View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Repositorio Institucional de la Universidad Autónoma del Estado de México

Universidad Autónoma del Estado de México

CENTRO UNIVERSITARIO UAEM TEXCOCO
INGENIERÍA EN COMPUTACIÓN

Comparación del desempeño de los Sistemas Gestores de Bases de Datos MySQL y PostgreSQL

TESIS

Que para obtener el Título de

Ingeniera en Computación

Presenta

López Herrera Patricia

Director de tesis
M en I.S.C. Irene Aguilar Juárez

Revisores de tesis
Dr. en ED Joel Ayala de la Vega
Dr. en C. Alfonso Zarco Hidalgo

Texcoco, Estado de México, a 18 de marzo del 2016

RESUMEN

La finalidad de este trabajo es realizar un estudio comparativo entre dos sistemas gestores de bases de datos, ambos con licencia libre: MySQL y PostgSQL.

Con la llegada del Internet, el software libre se ha consolidado como alternativa, técnicamente viable y económicamente sostenible al software comercial, contrariamente a lo que a menudo se piensa, convirtiéndose como otra alternativa para ofrecer los mismos servicios a un costo significativamente reducido, encontrando estas alternativas para gestores de Bases de Datos, se pretende determinar un soporte para recomendar la utilización de estos recursos de distribución libre, de la misma forma en que se confía en software con licencia o mucho mejor, para la toma de esta decisión se tendrán en cuenta factores como; seguridad en el almacenamiento de los datos, ventajas que proporciona un manejador de base de datos gratuito, volúmenes de información que soportara, cuál es su rendimiento, complejidad en la transacciones, consultas, concurrencias, soporte ofrecido, sistema de backup, confiabilidad ofrecida, considerando estos elementos como importantes para un buen desempeño en la aplicación y con esto el soporte de la decisión.

Se explica los aspectos y conceptos básicos de un SGB, iniciando con el origen y evolución de los SGBD, después se explican los diferentes modelos de SGBD en el mercado, posteriormente se analizan los factores que influyen en el buen desempeño, las ventajas y desventajas que los SGBD engloban. El éxito de las bases de datos relacionales en el procesamiento de datos se debe a la disponibilidad de lenguajes no procedurales los cuales pueden mejorar significativamente desarrollo de aplicaciones y la productividad del usuario final. La facilidad con la que se puede obtener información de las bases de datos suele determinar su valor para los usuarios. Los SGBD proporcionan un lenguaje especializado, denominado lenguaje de consultas, en el que se pueden formular las consultas y tienen mucho cuidado en evaluar las consultas de la manera más eficiente posible.

El manejo de transacciones es fundamental ya que hacen trasformaciones consistentes de los estados de un sistema preservando la consistencia del sistema. Una base de datos está en un estado consistente si obedece todas las restricciones de integridad definidas sobre ellas. Lo que

se persigue es por un lado tener transparencia adecuada de las acciones concurrentes a una base de datos y por otro lado tener una transparencia adecuada en el manejo de las fallas que se pueden presentar en una base de datos.

MySQL es la base de datos de código abierto más popular del mundo, se trata de un programa capaz de almacenar una enorme cantidad de datos de gran variedad y de distribuirlos para cubrir las necesidades de cualquier tipo de organización. MySQL incluye todos los elementos necesarios para instalar el programa, preparar diferentes niveles de acceso de usuario, administrar el sistema y proteger y hacer volcados de datos. Puede desarrollar sus propias aplicaciones de base de datos en la mayor parte de los lenguajes de programación utilizados en la actualidad y ejecutarlos en casi todos los sistemas operativos, MySQL utiliza el lenguaje de consulta estructurado (SQL). Se trata del lenguaje utilizado por todas las bases relacionales, este lenguaje permite crear bases de datos, así como agregar, manipular y recuperar datos en función de criterios específicos.

El Sistema Gestor de Bases de Datos Relacionales Orientadas a Objetos conocido como está derivado del paquete Postgres, es el gestor de bases de datos de código abierto más avanzado hoy en día, ofreciendo control de concurrencia multi-versión, soportando casi toda la sintaxis SQL (incluyendo subconsultas, transacciones, y tipos y funciones definidas por el usuario), contando también con un amplio conjunto de enlaces con lenguajes de programación (incluyendo C, C++, Java, perl, tcl y python). Y debido a la licencia liberal, PostgreSQL puede ser utilizado, modificado y distribuido por todo el mundo gratis de forma gratuita para cualquier fin, ya sea privada, comercial o académico.

CONTENIDO

LISTADO DE IMÁGENES

Listado de Tablas

INTRODUCCIÓN

El almacenamiento y tratamiento de los datos han sido el centro de atención de importantes aplicaciones para las organizaciones públicas, ya que se debe asegurar de que los recursos de sistemas y de información de una organización sean utilizados de la manera en que se decidió y que el acceso a la información allí contenida, así como su modificación sólo sea posible para las personas que se encuentren acreditadas y dentro de los límites de su autorización.

Los Sistemas Gestores de Bases de Datos (SGDB) son paquetes de software muy complejo y sofisticado. No se puede generalizar sobre los elementos que componen un SGBD ya que varían mucho unos de otros. Sin embargo, es muy útil conocer sus componentes y cómo se relacionan cuando se trata de comprender lo que es un sistema de bases de datos.

Es por eso que la mayoría de las organizaciones públicas buscan un SGBD que cumplan con un buen desempeño, que les convenga y a su vez que cumpla todos los requerimientos necesarios y sea lo menos vulnerable para así mantener la información de sus datos segura.

Los SGBD no siempre serán totalmente seguros ya que se ven amenazados por diversos factores y es necesario analizar y tener en cuenta cada módulo del SGBD tales como el procesador de las consultas, los registros de peticiones, los ficheros, el preprocesador del LMD. También es importante analizar los principales módulos del SGDB que son: su compilador, el control de autorización, el procesador de comandos, el optimizador de consultas, el gestor de transacciones, el planificador (scheduler), el gestor de recuperación y el gestor de buffers.

En este escrito se comparan el desempeño de dos Sistemas Gestores de Base de datos MySQL y PostgreSQL para conocer con mayor detalle las diferencias que existe entre ellos.

Capítulo 1
PLANTEAMIENTO DEL PROBLEMA

La mayoría de los datos sensibles del mundo están almacenados en sistemas gestores de bases de datos comerciales y atacar una base de datos es uno de los objetivos principales de muchas personas interesadas en la información de las organizaciones.

Los Sistemas Gestores de Bases de Datos (SGBD) no siempre cumplen con un buen desempeño y siempre estarán expuestos a muchos riesgos, tales como la incompatibilidad entre sistemas de acceso, la instalación del mismo, así como la necesidad de respaldar la información. El sistema gestor y la información almacenada en la base de datos deben estar protegida contra accesos no autorizados, destrucción o alteración con fines indebidos y la introducción accidental de inconsistencia así como también deben de tener un desempeño impecable y sin complicaciones.

Mientras que la atención generalmente se ha centrado en asegurar los perímetros de las redes, cada vez más las organizaciones se están enfocando en el desempeño que les brinda un buen sistema gestor para mantener la seguridad de sus datos protegidos contra cualquier amenaza.

Hoy ven en el software libre otra alternativa para ofrecer los mismos servicios a un costo, significativamente reducido, con el advenimiento de Internet, el software libre se ha consolidado como alternativa, técnicamente viable y económicamente sostenible al software comercial, contrariamente a lo que a menudo se piensa.

Hoy en día existen muchos SGBD dentro del mercado, esta documentación se enfoca específicamente a dos de ellos MySQL y PostgreSQL, ya que para algunas compañías es prioridad inminente la reducción de costos y usar los SGBD de manera efectiva.

Es por ello que nos planteamos ¿Qué sistema Gestor de Base de Datos tiene mejores estrategias de seguridad y desempeño MySQL o PostgreSQL para las organizaciones públicas?

1.1 JUSTIFICACIÓN

Los beneficios que se esperan obtener de este documento es tener información suficiente para poder evaluar un sistema gestor de base de datos con licencia libre, con la intensión de encontrar un SGBD totalmente seguro, que mantenga un desempeño eficaz y sin errores para que la integridad y seguridad en la información este a salvo y los usuarios estén confiados de que el SGBD no les cause ninguna dificultad.

La cantidad de información disponible crece cada vez más y para que los usuarios obtengan el máximo rendimiento de sus enormes y complejos conjuntos de datos son necesarias herramientas que simplifiquen las tareas de administrar los datos y de extraerlos en el momento preciso, confiados en que el sistema gestor no fallará y que sea el más adecuado brindándoles la confianza que necesitan.

Un Sistema Gestor de Base de Datos que brinde un buen desempeño trae consigo muchos beneficios a la organización, ya que su información siempre estará segura, disponible cuando se requiera, será autentica y confidencial desde su origen y estará respaldada en caso de que llegue a perderse, así como su rapidez en ejecución y a salvo de que cualquier persona puede tener acceso al SGDB. Aunque sabemos que hoy en día no hay seguridad absoluta, lo que se busca es un sistema gestor que reduzca el impacto de los riesgos a los que se expone.

Es esencial poder asegurar la calidad de los datos y de la información. En la calidad de una base de datos, se deben considerar tres aspectos: La calidad del SGBD (Sistema Gestor de Base de Datos)-relacional, orientado a objetos, objeto relacional, multidimensional, XML, etc.- que lo soporta, la calidad del modelo de datos (tanto conceptual, lógico como físico) y la calidad de los propios datos contenidos en la base de datos. (Piattini, García , Garzás, & Genero, 2008)

Lo que nos lleva a plantearnos ¿Qué sistema Gestor de Base de Datos tiene mejores estrategias de seguridad y desempeño MySQL o PostgreSQL para las organizaciones públicas?

1.2 OBJETIVOS

Evaluar las características de MySQL y PostgreSQL, haciendo un análisis comparativo para conocer cuál es el que presenta mejores propiedades y así tener una fuente de información para la toma de decisiones.

Objetivos Particulares

1. Identificar los criterios de evaluación para realizar el análisis comparativo.
2. Comparar MySQL y PostgreSQL usando los criterios antes seleccionados.

1.3 SUPUESTO

El desempeño del gestor de bases de datos Mysql es superior al gestor de bases de datos PostgreSQL.

1.4 Metodología

Investigación documental

- Fase 1. Identificar las características de los sistemas gestores de base de datos.
- Fase 2.- Identificar los criterios de desempeño de MySQL.
- Fase 3.- Identificar criterios de desempeño PosgreSQL.
- Fase 4.-Hacer una tabla comparativa sobre MySQL y PosgreSQL.
- Fase 5.-Reporte de resultados.

Capítulo 2
BASE DE DATOS.

El almacenamiento y tratamiento de los datos han sido el centro de atención de importantes aplicaciones para las organizaciones ya que se debe asegurar que los recursos de sistemas y de información de una organización sean utilizados de la manera en que se decidió y que el acceso a la información allí contenida, así como su modificación, sólo sea posible a las personas que se encuentren acreditadas y dentro de los límites de su autorización.

En este capítulo se explicaran los aspectos y conceptos básicos de un SGB, se inicia con una pequeña descripción del origen y evolución de los SGBD, después se explican los diferentes modelos de SGBD en el mercado, posteriormente se analizan los factores que influyen en el buen desempeño de los SGBD.

2.1 Orígenes de las bases de datos

El uso tedioso de papel para recoger datos y la lentitud de encontrar un dato concreto dio lugar al tratamiento automatizado de la información. Así surgen las bases de datos, brindando ahorro de espacio, velocidad de consulta y una disminución del manejo de papel.

Las bases de datos son aplicaciones informáticas destinadas al almacenamiento y la gestión de los datos que resuelven estos problemas y su evolución se ha llevado a largo de los años, como se explica en la tabla 1.

Tabla 1Evolución de las Bases de datos. (Pérez, 2010)

Años	
60´s y 70´s	Sistemas de ficheros y sistemas centralizados: un ordenador potente y terminales deficientes que accedían a los ficheros
80´s	Aparecen las bases de datos relacionales
80's y 90´s	Base de datos distribuidos, redes. Tecnología Cliente/Servidor. Un sistema de base de datos distribuidos se compone de un conjunto de sitios, conectados entre sí mediante algún tipo de comunicaciones.

2.2 Modelos de Bases de Datos.

Un modelo de datos es un conjunto de estructuras descriptivas de datos de alto nivel que oculta muchos detalles de almacenamiento de bajo nivel. Los SGBD (SBGD) permiten a los usuarios definir los datos que se van a almacenar en términos de un modelo de datos. La mayor parte de los sistemas actuales de gestión de bases de datos se basan en el modelo relacional de datos. (Pérez, 2010)

Aunque el modelo de datos del SGBD oculta muchos detalles, pese a todo está más cercano al modo en que el SGBD almacena los datos que al modo en que el usuario piensa en la empresa subyacente. Los **modelos semánticos de datos** son modelos de datos de alto nivel más abstractos que facilitan que los usuarios obtengan una buena descripción inicial de los datos de las empresas. Estos modelos contienen una amplia variedad de estructuras que ayudan a describir la situación real de las aplicaciones. No se pretende que los SGBD soporten directamente todas esas estructuras; suele crearse alrededor de un modelo de datos que tiene tan sólo unas cuantas estructuras básicas, como puede ser un modelo relacional. El diseño de bases de datos en términos de un modelo semántico sirve como útil punto de partida y se traduce posteriormente en el diseño de una base de datos en términos del modelo de datos que soporta realmente el SGBD. (Ramakrishnan & Gehrke, 2007)

Además del modelo de datos relacional (que se utiliza en numerosos sistemas, como DB2 de IBM, Informix, Oracle, Sybase, Access de Microsoft, FoxBase, Paradox, Tardem y Teardata) hay otros modelos de datos importantes, como el modelo jerárquico (que se emplea por ejemplo en IDS y en IDMS), el modelo orientado a objetos (que se usa por ejemplo, en Objectstore y en Versant) y el modelo relacional orientado a objetos (que se utiliza por ejemplo, en productos de SGBD de IBM, Informix, Objectstore, Oracle, Versant y otros). Aunque muchas bases de datos emplean los modelos jerárquicos y de red, y los sistemas basados en los orientados a objetos y relacional orientado a objetos están ganando aceptación en el mercado, el modelo dominante en la actualidad es el relacional.

En la tabla 2 se presentan algunos modelos de bases datos que surgieron en la época moderna (finales de 1960 - actualmente), así como sus principales características, lo que nos ayudará a elegir algún modelo cuando tengamos necesidad de usar una base de datos

Tabla 2 Cuadro comparativo de modelos de bases de datos.

	Jerárquico	Red	Relacional	Orientado a Objetos	Orientado a objetos Relacional
Características	Sus segmentos están dispuestos en forma de árbol. Los segmentos están enlazados mediante relaciones una a muchos	Un conjunto está formado en un solo registro propietario y uno o más	Se compone de varias tablas o relaciones. No pueden existir tablas con el mismo nombre.	Un objeto es una entidad del mundo real. Los objetos se identifican mediante un identificador de objetos, y es único para cada objeto.	Es una evolución desde el Modelo Relacional con conceptos del paradigma O.O. Define objetos dentro de otros objetos. Encapsula o asocia métodos con dichos objetos.
Ventajas	No ofrece ninguna posibilidad de definir las restricciones del usuario	Disminuye la redundancia Compartición de datos Aplicar restricciones de seguridad Mantiene la integridad.	Evita la duplicidad	Fácil manejo de tipos de datos complejos. Manipulación de datos complejos de toma rápida y ágil.	Permite relaciones que no cumplen la primera forma normal y permite la representa directa de las estructuras jerárquicas.
Relaciones	Presenta relaciones padre-hijo, es decir relaciones 1:N del modelo relacional pero las relaciones son unidireccionales	Registros se organizan en un conjunto de graficas arbitrarias. Pueden ser N:M, 1:N reflexiva, N:M reflexiva	Las relaciones son subconjuntos del producto cartesiano de la lista de dominios	Relaciones con multiplicidad 1. Relaciones con multiplicidad *. Relaciones n-arias	Las mismas del modelo O.O Relaciones anidadas.
Sistemas Manejadores de Bases de Datos	ADABAS FOCUS	CODASYL	SQL MYSQL ACCESS FOX PRO	ONTOS GEMSTONE	POSTGRES

Se describirá dos modelos de datos en este apartado: el modelo entidad-relación y el modelo relacional. Los diferentes modelos de datos que se han propuesto se clasifican en tres grupos

diferentes: modelos lógicos basados en objetos, modelos lógicos basados en registros y modelos físicos que se explicaran más adelante.

2.2.1 Modelo de datos entidad-relación (extendido)

El **modelo de datos entidad-relación** (E-R) está basado en una percepción del mundo real que consta de una colección de objetos básicos, llamados entidades, y de relaciones entre estos objetos. (Cares, s.f) ,

Una **entidad** es una «cosa» u «objeto» en el mundo real que es distinguible de otros objetos. Por ejemplo, cada persona es una entidad, y las cuentas bancarias pueden ser consideradas entidades. Las entidades se describen en una base de datos mediante un conjunto de atributos. Por ejemplo, los atributos número-cuenta y saldo describen una cuenta particular de un banco y pueden ser atributos del conjunto de entidades cuenta. Análogamente, los atributos nombre-cliente, calle-cliente y ciudad-cliente pueden describir una entidad cliente. (Silberschatz, 2007)

Un **atributo** extra, id-cliente, se usa para identificar unívocamente a los clientes (dado que puede ser posible que haya dos clientes con el mismo nombre, dirección y ciudad. Se debe asignar un identificador único de cliente a cada cliente.

Una **relación** es una asociación entre varias entidades. Por ejemplo, una relación impositor asocia un cliente con cada cuenta que tiene. El conjunto de todas las entidades del mismo tipo, y el conjunto de todas las relaciones del mismo tipo, se denominan respectivamente conjunto de entidades y conjunto de relaciones.

La estructura lógica general de una base de datos se puede expresar gráficamente mediante un diagrama ER, que consta de los siguientes componentes:

- **Rectángulos**, que representan conjuntos de entidades.
- **Elipses**, que representan atributos.
- **Rombos**, que representan relaciones entre conjuntos de entidades.
- **Líneas**, que unen los atributos con los conjuntos de entidades y los conjuntos de entidades con las relaciones.

Cada componente se etiqueta con la entidad o relación que representa, como se muestra en la imagen 1.

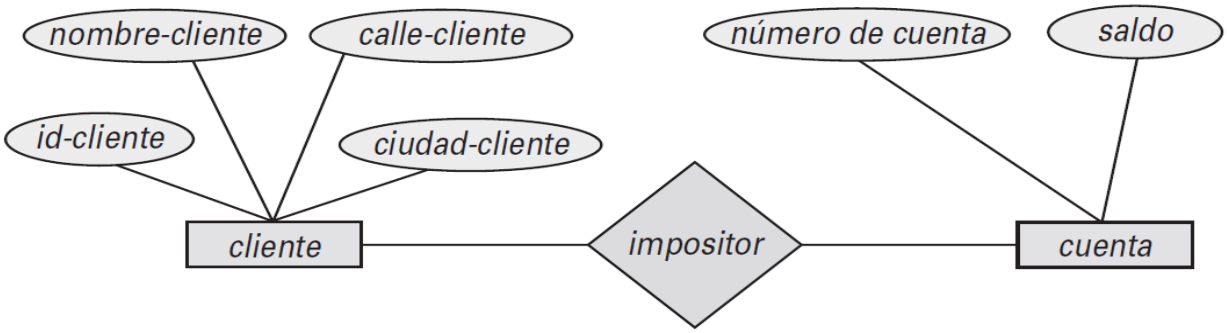

Imagen 1 Ejemplo de diagrama E-R.

El diagrama E-R indica que hay dos conjuntos de entidades cliente y cuenta, con los atributos descritos anteriormente. El diagrama también muestra la relación impositor entre cliente y cuenta.

Además de entidades y relaciones, el modelo E-R representa ciertas restricciones que los contenidos de la base de datos deben cumplir. Una restricción importante es la correspondencia de cardinalidades, que expresa el número de entidades con las que otra entidad se puede asociar a través de un conjunto de relaciones. Por ejemplo, si cada cuenta puede pertenecer sólo a un cliente, el modelo puede expresar esta restricción.

El modelo entidad-relación se utiliza habitualmente en el proceso de diseño de bases de datos.

2.2.2 Modelo relacional

La estructura central para la descripción de datos en este modelo son las relaciones, que se pueden considerar conjunto de registros.

La descripción de los datos en términos de un modelo de datos se denomina esquema. En el modelo relacional los esquemas de las relaciones especifican su nombre, el nombre de cada campo (o atributo o columna) y el tipo de cada campo. (Ramakrishnan & Gehrke, 2007)

En el modelo relacional se usa una colección de tablas para representar tanto los datos como las relaciones entre esos datos. Cada tabla tiene varias columnas, y cada columna tiene un nombre único. En la siguiente imagen basándonos en el ejemplo anterior del banco se presenta un

ejemplo de base de datos relacional consistente de dos tablas: una muestra los clientes del banco y la otra muestra las cuentas que pertenecen a esos clientes. La imagen 2 muestra, por ejemplo, que el cliente González, con número de DNI 19875430, vive en la calle Belgrano 1245 y tiene dos cuentas: C-101, con un saldo de 1.000 pesos, y C102, con un saldo de 15.000 pesos.

(Cares, s.f).

Cliente:

nombreCli	dniCli	direccionCli
González	24894732	Belgrano 1245
Gómez	19567831	Tucumán 395
López	24009258	Irigoyen 1260
Abraham	27894452	Dr. Ramón 300
Silva	25781034	Leloir 455
Repetto	19452859	Rivadavia 1100
Gómez	18825783	Santa Fe 1909
Pérez	21456092	Caviahue 245

Cuenta:

numeroCta	saldoCta
C-101	1.000
C-205	20.400
C-102	15.000
C-309	25.000
C-122	33.900
C-438	56.750
C-201	2.500
C-103	900
C-401	10.000

TieneCta:

dniCli	numeroCta
24894732	C-101
19567831	C-205
24894732	C-102
27894452	C-309
25781034	C-122
19452859	C-438
18825783	C-201
24009258	C-103
21456092	C-401

Imagen 2 Ejemplo Modelo Relacional

2.2.2.1 Manejo de Vistas

Una vista es una tabla cuyas filas no se almacenan en la base de datos de manera explícita, si no que se calculan cuando hace falta a partir de la definición de la vista. Considerando los niveles de abstracción. El esquema físico de las bases de datos relacionales describe el modo en que se almacenan las relaciones del esquema conceptual en término de la organización de archivos y de los índices empleados. El esquema conceptual es el conjunto de esquemas de las relaciones almacenadas en la base de datos. El mecanismo de vista proporciona, por tanto, el soporte para la independencia lógica en el modelo relacional. Es decir, se puede emplear para definir relaciones en el esquema externo que oculten a las aplicaciones las modificaciones del esquema conceptual de la base de datos. Por ejemplo, si se modifica el esquema de una relación almacenada, se puede definir una vista con el esquema antiguo y las aplicaciones que esperen ver el esquema antiguo podrán utilizar esa vista.

Las vistas también resultan valiosas en el contexto de la seguridad: se puede definir vistas que concedan acceso a cada grupo de usuarios solamente a la información que se les permite ver. (Ramakrishnan & Gehrke, 2007)

2.2.2.2 Actualización de vistas

La motivación subyacente al mecanismo de vistas es la personalización del modo en que los usuarios ven los datos. Los usuarios no deben tener que preocuparse por la distinción entre vistas y tablas base. Este objetivo se consigue realmente en el caso de las consultas a las vistas; las vistas se pueden utilizar igual que cualquier otra relación para la definición de consultas. No obstante, es natural desear especificar también las actualizaciones de las vistas, se debe tener presente que la norma SQL-92 solo permite que se especifiquen las actualizaciones en las vistas que se definan sobre una única tabla base y empleando solamente la selección y la proyección, sin utilizar operaciones de agregación. Estas vistas se denominan vistas de actualizables. (Tipos y modelos de bases de datos)

2.2.2.3 Necesidad de restringir las actualizaciones de las vistas.

Aunque las reglas de SQL sobre las vistas actualizables son más estrictas de lo necesario, hay algunos problemas fundamentales con las actualizaciones especificadas para las vistas que son una buena razón para limitar las clases de vistas que se pueden actualizar. (Ramakrishnan & Gehrke, 2007)

2.3 SGBD y Objetivos

Para usar bien un Sistema Gestor de Bases de Datos (SGBD), es necesario comprender también la manera en que funciona.

Los SGDB son aplicaciones software diseñadas para facilitar tareas. Al almacenar los datos en un SGBD en vez de un conjunto de archivos del sistema operativo, se pueden utilizar las características del SGBD para gestionar los datos de un modo robusto y eficiente. A medida que crecen el volumen de los datos y el número de usuarios (en las bases de datos corporativas actuales son habituales los centenares de gigabytes y los millares de usuario) el apoyo de los SGBD se vuelve indispensable. (Ramakrishnan & Gehrke, 2007)

Un SGBD Es el software que permite la utilización y/o la actualización de los datos almacenados en una (o varias) base(s) de datos por uno o varios usuarios desde diferentes puntos de vista y a la vez, se denomina sistema de gestión de bases de datos (SGBD). (Silberschatz, 2007) (Sistemas de Gestión de Bases de Datos)

2.3.1 Ventajas de los SGBD

El empleo de SGBD para gestionar los datos tiene muchas ventajas las cuales son: (Sistemas de Gestión de Bases de Datos)

- **Independencia con respecto a los datos.** Los programas de las aplicaciones no deben, en principio, exponerse a los detalles de la representación y el almacenamiento de los datos. El SGBD ofrece una vista abstracta de los datos que oculta esos detalles.
- **Acceso eficiente a los datos.** Los SGBD emplean gran variedad de técnicas sofisticadas para almacenar y recuperar los datos de manera eficiente. Esta característica resulta especialmente importante si los datos se guardan en dispositivos de almacenamiento externos.
- **Integridad y seguridad de los datos.** Si siempre se tienen acceso a los datos mediante el SGBD, éste puede hacer que se cumplan las restricciones de integridad. Por ejemplo antes de introducir la información salarial de un empleado, el SGBD puede comprobar que ni se haya superado el presupuesto del departamento. Además, puede hacer que se cumplan los controles de acceso que determinan los datos que son visibles para las diferentes clases de usuarios.
- **Administración de datos.** Cuando varios usuarios comparten los mismos datos, la centralización de la administración de esos datos puede ofrecer mejoras significativas. Los profesionales experimentados que comprenden la naturaleza de los datos que se gestionan y la manera en que los utilizan los diferentes grupos de usuarios, pueden responsabilizarse de la organización de la representación de los datos para minimizar la redundancia y mejorar el almacenamiento de los datos para hacer que la recuperación sea eficiente.
- **Acceso concurrente y recuperación en caso de fallo.** Los SGBD programan los accesos concurrentes a los datos de tal manera que los usuarios puedan creer que solo tienen

acceso a los datos un usuario a la vez. Además, los SGBD protegen a los usuarios de los efectos de los fallos del sistema.

- **Reducción del tiempo de desarrollo de las aplicaciones.** Los SGBD soportan funciones importantes que son comunes con muchas aplicaciones que tienen acceso a los datos del SGBD. Esto, junto con la interfaz de alto nivel con los datos, facilita el rápido desarrollo de aplicaciones. También es probable que las aplicaciones de los SGBD sean más robustas que las aplicaciones independientes similares, ya que el SGBD, maneja muchas tareas importantes.

Pero aun así dadas estas ventajas también existen razones para no utilizar un SGBD, como: (Sistemas de Gestión de Bases de Datos)

- Los SGBD son piezas de software complejas, optimizados para ciertos tipos de cargas de trabajo, y puede que su rendimiento no sea el adecuado para ciertas aplicaciones especializadas.
- Aplicaciones con estrictas restricciones de tiempo real o tan sólo unas pocas operaciones críticas bien definidas para las cuales hay que escribir un código a medida eficiente.
- Puede que una aplicación necesite manipular los datos de maneras no soportadas por el lenguaje de consultas. En esas situaciones, la vista abstracta de los datos ofrecida por el SGBD no coinciden con las necesidades de la aplicación y, en realidad, estorba.

En la mayor parte de las situaciones que requieren la gestión de datos a gran escala, no obstante, los SGBD se han convertido en una herramienta indispensable.

2.4 Objetivos SGBD

El objetivo fundamental de un SGBD consiste en suministrar al usuario las herramientas que le permitan manipular, en términos abstractos, los datos, o sea, de forma que no le sea necesario conocer el modo de almacenamiento de los datos en la computadora, ni el método de acceso empleado. (Silberschatz, 2007)

Los programas de aplicación operan sobre los datos almacenados en la base utilizando las facilidades que brindan los SGBD, los que, en la mayoría de los casos, poseen lenguajes

especiales de manipulación de la información que facilitan el trabajo de los usuarios. (Silberschatz, 2007)

Los SGDB brindan facilidad a la hora de elaborar tablas y establecer relaciones entre las informaciones contenidas en ellas. Pueden mantener la integridad de una base de datos permitiéndole a más de un usuario actualizar un registro al mismo tiempo y también puede impedir registros duplicados en una BD.

Lenguaje de Definición de Datos: (modifica la estructura o añade campos). La misión del **LDD** es describir y definir todos los esquemas que participen en la base de datos. Esto consiste en la descripción de los objetos que vamos a representar. La descripción de todas las estructuras que formen nuestra base de datos.

La función de LDD consiste en procesar las sentencias del lenguaje para identificar las descripciones de los distintos elementos de los esquemas y almacenar la descripción del esquema en el catálogo o diccionario de datos.

Existen cuatro operaciones básicas:

- CREATE
- ALTER
- DROP
- TRUNCATE

Manipulación de datos: **LMD** recoge toda la operación de intercambio de datos entre las tablas, estas operaciones pueden ser de consulta o de puesta al día (inserción, modificación, supresión) estas operaciones se realizan con la ayuda del denominado LMD (Lenguaje de Manipulación de Datos.)

Los LMD Se clasifican en dos grandes grupos de:

- Lenguajes de consulta procedimentales

Lenguajes procedimentales. En este tipo de lenguaje el usuario da instrucciones al sistema para que realice una serie de procedimientos u operaciones en la base de datos para calcular un resultado final.

- Lenguajes de consulta no procedimentales

En los lenguajes no procedimentales el usuario describe la información deseada sin un procedimiento específico para obtener esa información.

Las principales características de los lenguajes no procedimentales son:

- Consultas
- Operaciones
- Insertar datos
- Modificar datos
- Suprimir datos

2.5 Niveles de abstracción de los SGBD

Los datos en los SGBD se describen en tres niveles de abstracción, como se puede ver en la Imagen 3. La descripción de las bases de datos consta de un esquema en cada uno de esos tres niveles de abstracción: conceptual, físico y externo

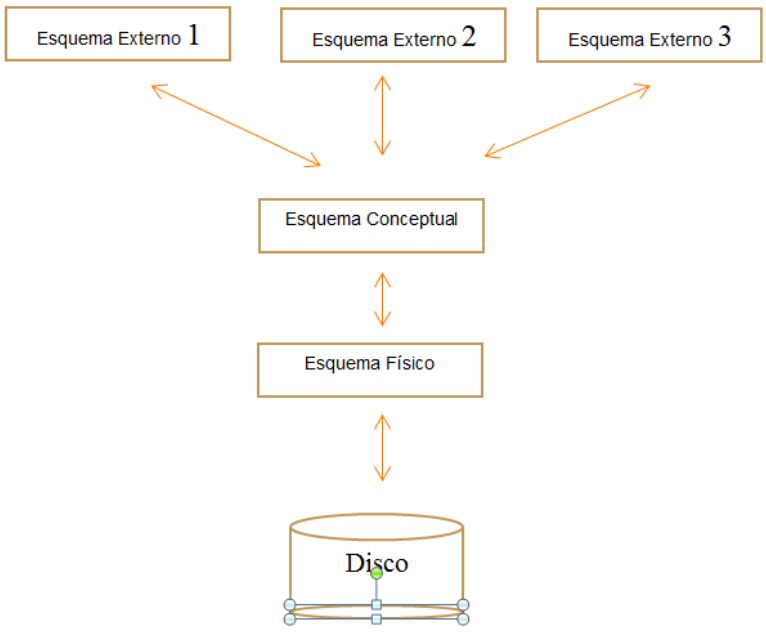

Imagen 3 Niveles de abstracción en los SGBD

El esquema conceptual (denominado varias veces esquema lógico) describe los datos almacenados en términos del modelo de datos de SGBD. En un SGBD relacional, el esquema conceptual describe todas las relaciones almacenadas en la base de datos.

Esquema Físico

El esquema físico especifica detalles adicionales del almacenamiento. Esencialmente, el esquema físico resume el modo en que las relaciones descritas en el esquema conceptual se guardan realmente en dispositivos de almacenamiento secundario como discos y cintas.

Es necesario decidir la organización de archivos que se va a utilizar para guardar las relaciones y crear estructuras de datos auxiliares, denominadas índices, para acelerar las operaciones de recuperación.

Las decisiones sobre el esquema físico se basan en conocer el modo en que suele tener acceso a los datos.

Esquema Externo

Los esquemas externos, que suelen serlo también en términos del modelo de datos del SGBD, permiten personalizar (y autorizar) el acceso a los datos a los usuarios y grupos de ellos. Cualquier base de datos tiene exactamente un esquema conceptual y un esquema físico, porque solo tiene guardado un conjunto de relaciones, pero puede tener varios esquemas externos, cada uno de ellos adaptados a un grupo de usuarios concreto. Cada esquema externo consiste en un conjunto de una o varias vistas y relaciones del esquema conceptual. Una vista es, conceptualmente, una relación pero los registros de la vista no se guardan en el SGBD. El diseño del esquema externo se guía por las necesidades del usuario final. (Ramakrishnan & Gehrke, 2007)

2.6 Independencia con respecto a los datos

Una ventaja muy importante del empleo de un SGBD es que ofrece **independencia con respecto a los datos**. Es decir, los programas de aplicación quedan aislados de las modificaciones debido al modo en que se estructuran y se guardan en los datos. La independencia con respecto a los datos se consiguen mediante el empleo de los tres niveles de

abstracción de los datos; en concreto, el esquema conceptual y el externo ofrecen claras ventajas en este campo.

Las relaciones en el esquema externo (relaciones de vistas) se generan, en principio, a petición de los usuarios a partir de las relaciones correspondientes del esquema conceptual. Si se modifica el esquema conceptual, se puede modificar la definición de la relación de vistas para que esa relación se siga calculando como antes.

Se puede proteger a los usuarios de las modificaciones en la estructura lógica de los datos, o en la elección de las relaciones que se guardan. Esta propiedad se denomina independencia con respecto a los datos.

A su vez, el esquema conceptual aísla a los usuarios respecto de las modificaciones en los detalles de almacenamiento físico. Esta propiedad se conoce independencia física con respecto a los datos. El **esquema conceptual** oculta detalles como el modo en que se disponen realmente datos en el disco, la estructura de los archivos y la elección de los índices. En tanto el esquema conceptual siga siendo el mismo, se pueden modificar esos detalles de almacenamiento sin alterar las aplicaciones. (Ramakrishnan & Gehrke, 2007)

2.7 Características de los SGBD

En este módulo se analizara qué características son deseables en los SBGD y qué capacidades deben ofrecer.

La particularidad con los que debe contar los SGBD son:

- Permite **crear y gestionar** base de datos de forma fácil, cómoda y rápida.
- Ofrece una gran **flexibilidad** para el trabajo con base de datos relacionales.
- Ofrece **un ambiente agradable** dado por su interfaz gráfica.
- **Abstracción de la información**. Los SGBD ahorran a los usuarios detalles acerca del almacenamiento físico de los datos. Da lo mismo si una base de datos ocupa uno o cientos de archivos, este hecho se hace transparente al usuario. Así, se definen varios niveles de abstracción.

- **Independencia.** La independencia de los datos consiste en la capacidad de modificar el esquema (físico o lógico) de una base de datos sin tener que realizar cambios en las aplicaciones que se sirven de ella.
- **Redundancia mínima**. Un buen diseño de una base de datos logrará evitar la aparición de información repetida o redundante. De entrada, lo ideal es lograr una redundancia nula; no obstante, en algunos casos la complejidad de los cálculos hace necesaria la aparición de redundancias.
- **Consistencia**. En aquellos casos en los que no se ha logrado esta redundancia nula, será necesario vigilar que aquella información que aparece repetida se actualice de forma coherente, es decir, que todos los datos repetidos se actualicen de forma simultánea.
- **Seguridad**. La información almacenada en una base de datos puede llegar a tener un gran valor. Los SGBD deben garantizar que esta información se encuentra seguridad frente a usuarios malintencionados, que intenten leer información privilegiada; frente a ataques que deseen manipular o destruir la información; o simplemente ante las torpezas de algún usuario autorizado pero despistado. sistema de permisos a usuarios y grupos de usuarios, que permiten otorgar diversas categorías.
- **Integridad.** Se trata de adoptar las medidas necesarias para garantizar la validez de los datos almacenados. Es decir, se trata de proteger los datos ante fallos de hardware, datos introducidos por usuarios descuidados, o cualquier otra circunstancia capaz de corromper la información almacenada.
- **Respaldo y recuperación**. Los SGBD deben proporcionar una forma eficiente de realizar copias de respaldo de la información almacenada en ellos, y de restaurar a partir de estas copias los datos que se hayan podido perder.
- **Control de la concurrencia**. En la mayoría de entornos (excepto quizás el doméstico), lo más habitual es que sean muchas las personas que acceden a una base de datos, bien para recuperar información, bien para almacenarla. Y es también frecuente que dichos accesos se realicen de forma simultánea. Así pues, un SGBD debe controlar este acceso concurrente a la información, que podría derivar en inconsistencias.

2.8 Componentes de un sistema de gestión de bases de datos

Los SGBD son paquetes de software muy complejo y sofisticado que deben proporcionar los servicios comentados en el capítulo anterior. No se puede generalizar sobre los elementos que componen un SGBD ya que varían mucho unos de otros. Sin embargo, es muy útil conocer sus componentes y cómo se relacionan cuando se trata de comprender lo que es un sistema de bases de datos.

Un SGBD tiene varios módulos, cada uno de los cuales realiza una función específica. El sistema operativo proporciona servicios básicos al SGBD, que es construido sobre él.

El procesador de consultas es el componente principal de un SGBD. Transforma las consultas en un conjunto de instrucciones de bajo nivel que se dirigen al gestor de la base de datos.

El gestor de la base de datos es el interface con los programas de aplicación y las consultas de los usuarios. El gestor de la base de datos acepta consultas y examina los esquemas externo y conceptual para determinar qué registros se requieren para satisfacer la petición. Entonces el gestor de la base de datos realiza una llamada al gestor de ficheros para ejecutar la petición.

El gestor de ficheros maneja los ficheros en disco en donde se almacena la base de datos. Este gestor establece y mantiene la lista de estructuras e índices definidos en el esquema interno. Si se utilizan ficheros dispersos, llama a la función de dispersión para generar la dirección de los registros. Pero el gestor de ficheros no realiza directamente la entrada y salida de datos. Lo que hace es pasar la petición a los métodos de acceso del sistema operativo que se encargan de leer o escribir los datos en el buffer del sistema.

El preprocesador del LMD convierte las sentencias del LMD embebidas en los programas de aplicación, en llamadas a funciones estándar escritas en el lenguaje anfitrión. El preprocesador del LMD debe trabajar con el procesador de consultas para generar el código apropiado.

El compilador del LDD convierte las sentencias del LDD en un conjunto de tablas que contienen metadatos. Estas tablas se almacenan en el diccionario de datos.

El gestor del diccionario controla los accesos al diccionario de datos y se encarga de mantenerlo. La mayoría de los componentes del SGBD acceden al diccionario de datos.

2.8.1 Los principales componentes

En opinión de (Silberschatz, 2007) los diferentes componentes de los SGBD son los siguientes:

- **Control de autorización.** Este módulo comprueba que el usuario tiene los permisos necesarios para llevar a cabo la operación que solicita.
- **Procesador de comandos.** Una vez que el sistema ha comprobado los permisos del usuario, se pasa el control al procesador de comandos.
- **Control de la integridad.** Cuando una operación cambia los datos de la base de datos, este módulo debe comprobar que la operación a realizar satisface todas las restricciones de integridad necesarias.
- **Optimizador de consultas.** Este módulo determina la estrategia óptima para la ejecución de las consultas.
- **Gestor de transacciones.** Este módulo realiza el procesamiento de las transacciones.
- **Planificador (scheduler).** Este módulo es el responsable de asegurar que las operaciones que se realizan concurrentemente sobre la base de datos tienen lugar sin conflictos.
- **Gestor de recuperación.** Este módulo garantiza que la base de datos permanece en un estado consistente en caso de que se produzca algún fallo.
- **Gestor de buffers**. Este módulo es el responsable de transferir los datos entre memoria principal y los dispositivos de almacenamiento secundario. A este módulo también se le denomina gestor de datos.

2.9 Funciones SGBD

En opinión de (Silberschatz, 2007) existen diferentes funciones que deben de cumplir los SGDB, son los siguientes:

- **Definición de los Datos.** El SGBD debe ser capaz de aceptar las definiciones de datos en versión fuente y convertirlas en la versión objeto. El SGBD debe incluir componentes procesadores para cada uno de los lenguajes de definición de datos

- **Manipulación de los Datos.** El SGBD debe atender las solicitudes de los usuarios para extraer, actualizar, adicionar o suprimir datos. El SGBD debe incluir un componente procesador del Lenguaje de manipulación de datos
- **Seguridad e Integridad de los Datos.** El SGBD debe supervisar las solicitudes de los usuarios y rechazar los intentos de violar las medidas de seguridad e integridad definidas por el Administrador de la Base de Datos DBA.
- **Recuperación y concurrencia de los datos.** El principal objetivo de la implantación de una base de datos es poner a disposición de un gran número de usuarios en conjunto integrado de datos, estos datos podrán ser manipulados por los diferentes usuarios y es ahora cuando se debe garantizar la coherencia de los datos después de las diversas manipulaciones.
- **Diccionario de Datos.** Es un listado organizado de todos los datos que pertenecen a un sistema. Su objetivo es dar precisión sobre los datos que se manejan.
- **Desempeño**. El SGBD debe ejecutar todas las funciones en la forma eficiente.

2.10 Tipos de Sistemas Gestores de Bases de Datos.

La clasificación de los SGBD se hace en función de los criterios de uso. El criterio más específico es según el modelo de base de datos, jerárquica, de red, relacional y orientada a objetos.

La clasificación de los SGBD, se ve representada en la Imagen 4.

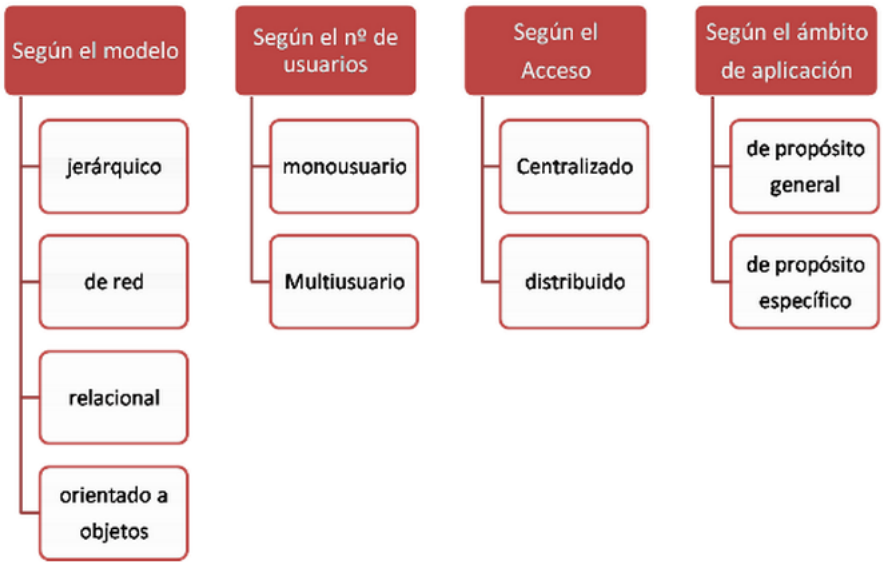

Imagen 4 Clasificación de Gestores de Bases de Datos.

2.11 La seguridad en los SGBD

Las amenazas son aquellas personas, gobiernos, compañías u otras organizaciones, de las que uno quisiera proteger su sistema. Las amenazas pueden causar daño tanto accidental como intencionadamente. El rango de los daños que puede originar una amenaza consiste en: borrar los datos, alterar los datos en forma indetectable, leer los datos para comprometer la posición de la organización vulnerada y destruir el sistema.

Determinar de qué amenazas hay que preocuparse depende de la clase de datos que haya que proteger. El administrador de una base de datos o el gestor de la seguridad de una organización, uno otorga acceso o restringe privilegios a los usuarios implicados en la base de datos. (Theriault & Newman, 2002)

La integridad de un sistema es un tópico amplio que contempla la calidad y la solidez de tal sistema. Existen muchos medios de violar la integridad de un sistema, por lo que para conservarla intacta es menester asegurarse de que los atacantes no pueden denegar acceso s otros usuarios, corromper los datos, corromper las instrucciones e, incluso, destruir el sistema.

La autentificación es el proceso de identificar adecuadamente a un usuario. Existen muchos medios inteligentes de hacerlo, si bien ninguno es perfecto y cada uno presenta sus puntos débiles. Para un proveedor de software, el mejor método de asegurarse de que no hay puntos débiles en seguridad, es hacer disponible su implantación públicamente. Si existe alguna dificultad de implantación, la gente encontrara el punto débil y se lo hará saber. (Zevallos)

2.11.1 Contraseñas

La forma más común de autentificarse uno mismo ante la seguridad de una máquina, es utilizar una contraseña. No obstante, la aplicación de contraseñas puede ser distinta de un sistema a otro. En el caso más sencillo se utiliza una única contraseña para autentificar a todos los usuarios del

sistema. Es claro que la seguridad de un mecanismo así es muy débil. La mayor parte de los sistemas de un mecanismo de contraseñas más complejos, que requieren una contraseña distinta para cada usuario. Otro requisito es que dicha contraseña no se almacene como texto llano en la máquina.

Desde el punto de vista de la base de datos, es necesario asegurar que solamente los usuarios legales y autorizados pueden interactuar con la base de datos y su contenido. Disponiendo de potestad, tanto para permitir su acceso como para auditar las actividades de los usuarios, es posible garantizar que solamente las personas y autentificadas gozan de acceso al sistema que compone la red. Parte de la seguridad de la base de la base de datos consiste en examinar las cuentas que se originan automáticamente cuando creamos nuestra base de datos. Es vitalmente importante que conozcamos para que sirve cada una de todas esas cuentas y si es necesario mantenerlas como son. O modificarles el nombre de usuario y la contraseña, o eliminarlas completamente del sistema.

2.12 ¿En qué situaciones la información de la base de datos se ve vulnerable?

¿Cómo se puede perder datos de su sistema o de su base de datos?

- **Error de usuario**. Cuando el usuario borra accidentalmente un registro, o un grupo de registros, equivocado.
- **Error del programador.** Un programador escribe un código para borrar información y este código funciona de manera distinta en desarrollo y prueba, a como lo hace en producción.
- **Error del administrador**. El administrador de una base de datos cree erróneamente que está borrando una tabla en la base de datos de desarrollo, pero realmente lo está haciendo en la de producción.
- **Error debido a un hacker**. Un hacker entra en un sistema y borra malintencionadamente.
- **Error de hardware.** Un equipo del sistema, como un disco duro o un manipulador de datos, se avería y, o bien no se puede extraer los datos, o bien están corrompidos.

- **Errores por parte de la naturaleza.** Un incendio o una inundación, sepulta el centro de datos en que se hallan los equipos en cuestión.

Los cuatro primeros puntos anteriores se deben directa o indirectamente a errores humanos asociados a errores de software y los dos últimos a causas fuera del control de usuario.

Para garantizar la seguridad de los datos de una compañía hay que estar preparado para todas y cada una de las situaciones de emergencia listadas anteriormente.

Características de gestión de contraseñas.

La capacidad de definir las características de las contraseñas ha sido una faceta de muchos sistemas durante años. Las contraseñas ofrecen la primera línea de defensa contra el acceso no autorizado al equipo. Cuanto más segura sea la contraseña, más protegido estará el equipo contra hackers y software malintencionado entre otros factores. Una contraseña segura debe de contar con las siguientes características:

- **Composición y complejidad**: Definir la longitud de la contraseña y los caracteres, números y signos de puntuación que hay que utilizar al formar la contraseña.
- **Envejecimiento y caducidad**: Determinar la edad que puede tener una contraseña antes de tener que cambiarla.
- **Historia**: seguimiento de las contraseñas que han existido ya en un plazo especifico de tiempo y numero de cambios de contraseña para garantizar que no se utiliza la misma contraseña demasiado frecuentemente.
- **Bloqueo**: Definir si hay que bloquear o desbloquear, manualmente o automáticamente, una cuenta y cuándo.

La mayoría de los sistemas disponen de un mecanismo para definir el plazo de validez de una contraseña en el sistema. Cuando expira el plazo de validez. La contraseña caduca y el usuario ha de aportar una contraseña nueva. (Theriault & Newman, 2002)

Capítulo 3 PROCESAMIENTO DE CONSULTAS

El éxito de las bases de datos relacionales en el procesamiento de datos se debe a la disponibilidad de lenguajes no procedurales los cuales pueden mejorar significativamente desarrollo de aplicaciones y la productividad del usuario final. Ocultando los detalles de bajo nivel acerca de la localización física de datos, los lenguajes de bases de datos relacionales permiten la expresión consultas complejas en una forma concisa y simple. Particularmente, para construir la respuesta a una consulta, el usuario no tiene que especificar de manera precisa el procedimiento que se debe seguir. Este procedimiento es llevado a cabo por un módulo del SGBD llamado el procesador de consultas

La facilidad con la que se puede obtener información de las bases de datos suele determinar su valor para los usuarios. Los SGBD proporcionan un lenguaje especializado, denominado lenguaje de consultas, en el que se pueden formular las consultas. Una característica muy atractiva del modelo relacional es que soporta lenguajes de consulta potentes. El cálculo relacional es un lenguaje de consultas formal basado en la lógica matemática, y las consultas en ese lenguaje tienen un significado preciso e intuitivo. El álgebra relacional es otro lenguaje formal de consultas, basado en un conjunto de operadores para la manipulación de las relaciones, que es equivalente en potencia al cálculo relacional.

Los SGBD tienen mucho cuidado en evaluar las consultas de la manera más eficiente posible. Por supuesto, la eficiencia de la evaluación de las consultas viene determinada en gran parte por la manera en que los datos se han almacenado físicamente. Se pueden utilizar índices para acelerar muchas consultas. Los SGBD permiten a los usuarios crear, modificar, y consultar datos mediante los lenguajes de manipulación de datos (LMD). Por tanto el lenguaje de consultas es tan solo una parte del LMD, que también proporciona estructuras para añadir, eliminar y modificar datos. En este capítulo revisaremos el procesamiento de consultas en base de datos

3.1 El problema de procesamiento de consultas.

La función principal de un procesador de consultas relacionales es transformar una consulta en una especificación de nivel, típicamente calculo relacional, a una consulta equivalente de unas especificaciones de bajo nivel, típicamente alguna variación del algebra relacional. La consulta de bajo nivel implementa de hecho la estrategia de ejecución para la consulta. La transformación debe ser correcta y eficiente. Es correcta si la consulta de bajo nivel tiene la misma semántica que la consulta original, esto es, si ambas consultas producen el mismo resultado. Sin embargo, producir una estrategia de ejecución eficiente es mucho más complicado. Una consulta en el cálculo relacional puede tener muchas transformaciones correctas y equivalentes en algebra relacional. Ya que cada estrategia de ejecución equivalente puede conducir a consumos de recursos de cómputo diferentes, la dificultad más importante es seleccionar la estrategia de ejecución que minimiza el consumo de recursos.

3.2 Objetivos de la optimización de consultas.

El objetivo del procesamiento de consultas es transformar una consulta sobre una base de datos en una especificación de alto nivel a una estrategia de ejecución eficiente expresada en un lenguaje de bajo nivel sobre bases de datos locales.

3.3 La complejidad de las operaciones del álgebra relacional

La complejidad de las operaciones del algebra relacional afectan directamente su tiempo de ejecución establecen algunos principios útiles al procesador de consultas. Esos principios pueden ayudar en elegir estrategias de ejecución final. La forma más simple de definir la complejidad es en términos de cardinalidad de las relaciones independientemente de los detalles de implementación tales con fragmentación y estructuras de almacenamiento.

En la Tabla 3 se presenta la complejidad de operaciones unitarias y binarias en el orden creciente de complejidad de las operaciones del algebra lineal.

Tabla 3Complejidad de las operaciones del álgebra relacional.

Operación	Complejidad
Selección Proyección (Sin eliminación de duplicados)	$O(n)$
Proyección (Con eliminación de duplicados) Agrupación	$O(n * \log\ n)$
Junta Semijunta División Operadores sobre conjuntos	$O(n * \log\ n)$
Productos cartesianos	$O(n^2)$

Esta mirada a la complejidad de las operaciones sugiere dos principios:

1. Las operaciones más selectivas que reducen las cardinalidades deben ser ejecutadas primero.
2. Las operaciones deben ser ordenadas en el orden de complejidad creciente de manera que el producto cartesiano puede ser evitado, o al menos ejecutado al final de la estrategia.

3.4 Caracterización de los procesadores de consultas.

Mencionando algunas características importantes de los procesadores de consultas.

- **Tipo de optimización.**

El problema de optimización de consultas es altamente demandante en tiempo de ejecución. Así existen dos estrategias para su solución: búsqueda exhaustiva o el uso de heurísticas. Los algoritmos de búsqueda exhaustiva tienen una complejidad combinatoria en número de relaciones de consulta. Obtienen la transformación optima, pero se aplican a sus consultas simples dado su tiempo de ejecución.

Por otro lado, los algoritmos heurísticos obtienen solo aproximaciones a la transformación óptima pero lo hacen en un tiempo de ejecución razonable. Las heurísticas más directas a aplicar son el agrupamiento expresiones comunes para evitar el cálculo repetido de las mismas, aplicar primero las operaciones selección y proyección, reemplazar una junta por una serie de semijuntas y reordenar operaciones para reducir el tamaño de las relaciones intermedias.

- **Granularidad de la optimización.**

Existen dos alternativas: considerar solo una consulta a la vez o tratar de optimizar múltiples consultas. La primera alternativa no considera el uso de resultados comunes intermedios. En el segundo caso puede obtener transformaciones eficientes si las consultas son similares. Sin embargo, el espacio de decisión es mucho más amplio lo que afecta grandemente el tiempo de ejecución de la optimización.

- **Tiempo de optimización.**

Una consulta puede ser optimizada en tiempos diferentes con relación a tiempo de ejecución de consulta. La optimización se puede realizar de manera estática antes de ejecutar la consulta o de forma dinámica durante la ejecución de la consulta. La optimización estática se hace en tiempo de compilación de la consulta. Así, el costo de la optimización puede ser amortizada sobre múltiples ejecuciones de la misma consulta.

Un tercer enfoque, conocido como hibrido, utiliza básicamente un enfoque estático, pero se puede aplicar un enfoque dinámico cuando los tamaños de las relaciones estimados están alejados de los tamaños actuales.

- **Estadísticas.**

La efectividad de una optimización recae en las estadísticas de la base de datos. La optimización dinámica de consultas requiere de estadísticas para elegir las operaciones que deben realizarse primero. La optimización estática es aún más demandante ya que el tamaño de las relaciones intermedias también debe ser estimado basándose en estadísticas.

- **Nodos de Decisión.**

Cuando se utiliza la optimización estática, un solo nodo o varios de ellos pueden participar en la selección de la estrategia a ser aplicada para ejecutar la consulta.

3.5 Manejo de transacciones.

Los SGBD deben proteger a los usuarios de los efectos de los fallos del sistema asegurando que todos los datos (y el estado de las aplicaciones activas) vuelvan a un estado consistente cuando se reinicie el sistema tras un fallo (Ramakrishnan & Gehrke, 2007)

Qué pasa cuando, por ejemplo dos consultas tratan de actualizar el mismo elemento de datos, o si ocurre una falla en el sistema durante la ejecución de la consulta. Dada la naturaleza de una consulta, de lectura o actualización, a veces no se puede simplemente reactivar la ejecución de una consulta, puesto que algunos datos pueden haber sido modificados antes de la falla. El no tomar en cuenta esos factores puede conducir a que la información en la base de datos contenga datos incorrectos.

El concepto fundamental aquí es la noción de ¨ejecución consistente¨ o ¨procesamiento confiable ¨ asociada con el concepto de una consulta. El concepto de una transacción es usado dentro del dominio de la base de datos como una unidad de cómputo consistente y confiable.

Una **transacción** es una colección de acciones que hacen trasformaciones consistentes de los estados de un sistema preservando la consistencia del sistema. Una base de datos está en un estado consistente si obedece todas las restricciones de integridad definidas sobre ellas.

Se busca asegurar que la base de datos nunca entre en un estado de inconsistencia. Sin embargo, durante la ejecución de una transacción, la base de datos puede estar temporalmente en un estado de inconsistente.

Lo que se persigue es el manejo de transacciones es por un lado tener transparencia adecuada de las acciones concurrentes a una base de datos y por otro lado tener una transparencia adecuada en el manejo de las fallas que se pueden presentar en una base de datos.

- **Condiciones de terminación de una transacción**

Una transacción siempre termina, aun en la presencia de fallas. Si una transacción termina de manera exitosa se dice que la transacción hace un commit. Si la transacción es abortada, su ejecución es detenida y todas sus acciones ejecutadas hasta el momento son deshechas.

- **Caracterización de transacciones.**

Los elementos de datos que lee una transacción se dice que constituyen el conjunto de lectura (RS). Similarmente, los elementos de datos que una transacción escribe se les denominan el conjunto de escritura (ws).

- **Propiedades de las transacciones**

Las transacciones proporcionan una ejecución atómica y confiable en presencia de fallas, una ejecución correcta en presencia de accesos de usuarios múltiples y un manejo correcto de réplicas.

Se denomina **ACID** (Atomicidad, Consistencia, Aislamiento y Durabilidad) a las características que permiten clasificar las transacciones de los SGBD. Cuando se dice que es ACID compliant se indica que éste permite realizar transacciones.

Las propiedades ACID de una transacción son las siguientes:

1. **Atomicidad**. Se refiere al hecho de que una transacción se trata como una unidad de operación. Por lo tanto, o todas las acciones de la transacción se realizan o ninguna de ellas se lleva a cabo. La atomicidad requiere que si una transacción se interrumpe por una falla, sus resultados parciales deben ser deshechos. La actividad referente a preservar la atomicidad de transacciones en presencia de abortos debido a errores de entrada, sobrecarga del sistema o interbloqueo se le llama recuperación de transacciones. La actividad de asegurar la atomicidad en presencia de caídas del sistema se le llama recuperación de caídas.

2. **Consistencia**. La consistencia de una transacción es simplemente su correctitud. En otras palabras. Una transacción es un programa correcto que lleva la base de datos de un estado consistente a otro con la misma característica. Debido a esto, las transacciones no violan las restricciones no violan las restricciones de integridad de una base de datos.

3. **Aislamiento**. Una transacción en ejecución no puede revelar sus resultados a otras transacciones concurrentes antes de su commit. Más aún, si varias transacciones se ejecutan concurrentemente, los resultados deben ser los mismos que si ellas se hubieran ejecutado de manera secuencial (seriabilidad).

4. **Durabilidad**. Es la propiedad de las transacciones que asegura, que una vez que una transacción hace su commit, sus resultados son permanentes y no pueden ser borrados de la base de datos. Por lo tanto los SGBD aseguran que los resultados de una transacción sobrevivirán a fallas del sistema. Esta propiedad motiva el aspecto de recuperación de base de datos, el cual trata sobre cómo recuperar la base de datos aun estado consistente en donde todas las acciones que han hecho commit queden reflejadas.

3.6 Ejecución concurrente de las transacciones.

Una importante tarea de los SGBD es la programación de los accesos concurrentes a los datos de modo que cada usuario pueda ignorar con seguridad el hecho de que otros tienen acceso a los datos de manera concurrente. La importancia de esta tarea no puede subestimarse, ya que las bases de datos suelen estar compartidas por gran número de usuarios, que remiten sus consultas al SGBD de manera independiente y, sencillamente, no puede esperarse que afronten las modificaciones arbitrarias que realicen otros usuarios de manera concurrente. El SGBD permite que los usuarios crean que sus programas se ejecutan aisladamente, uno tras otro en el orden escogido por el SGBD.

Los protocolos de bloqueo son conjuntos de reglas que debe seguir cada transacción (y que el SGBD debe hacer que se cumplan) para garantizar que, aunque las acciones de varias transacciones se intercalen, el efecto neto sea idéntico a la ejecución de todas las transacciones en un orden consecutivo dado. Un bloqueo es un mecanismo empleado para controlar el acceso a los objetos de la base de datos. Los SGBD suelen soportar dos tipos de bloqueos: los compartidos sobre un objeto pueden establecerlos dos transacciones diferentes al mismo tiempo, pero el bloqueo exclusivo de un objeto garantiza que ninguna otra transacción establezca un bloqueo sobre este objeto. (Ramakrishnan & Gehrke, 2007)

3.6.1 Las transacciones no completadas y los fallos del sistema

El control de concurrencia trata con los problemas de aislamiento y consistencia del procesamiento de transacciones.

Si no se hace un adecuado control de concurrencia, se pueden presentar dos anomalías. En primer lugar, se pueden perder actualizaciones provocando que los efectos de algunas transacciones no se reflejen en la base de datos. En segundo término, pueden presentarse recuperaciones de información inconsistentes.

Las transacciones pueden interrumpirse antes de ser completadas por gran variedad de motivos como, por ejemplo un fallo en el sistema. Los SGBD deben garantizar que las modificaciones llevadas a cabo por esas transacciones no completadas se eliminen de las bases de datos. Los SGBD mantienen un registro de todas las operaciones de escritura en la base de datos. Una propiedad de ese registro es que cada acción de escritura debe registrarse en el registro (disco) antes de que la modificación correspondiente se refleje en la propia base de datos, en caso contrario, si el sistema fallase justo tras la realización de la modificación de la base de datos pero antes de que esta se reflejara en el registro, el SGBD no sería capaz de detectarla y deshacerla, a esta propiedad se le denomina registro de escritura previa (Write-Ahead Log, WAL). Para garantizarla, el SGBD debe poder obligar de manera selectiva a que se guarde en el disco una página que se halle en la memoria.

El registro se utiliza para garantizar que las modificaciones llevadas a cabo por las transacciones completadas con éxito no se pierdan debido a fallos del sistema. La devolución de la base de datos a un estado consistente tras un fallo del sistema puede ser un proceso lento, ya que el SGBD debe garantizar que el efecto de todas las transacciones que se completaron antes del fallo se restaure, y que el de las transacciones no completadas se deshaga. El tiempo necesario para recuperarse de un fallo puede reducirse obligando de manera periódica que cierta información se guarde en disco; esas operaciones periódicas se denominan puntos de revisión.

Capítulo 4 MySQL

MySQL es la base de datos de código abierto más popular del mundo. Código abierto significa que todo el mundo puede acceder al código fuente, es decir, al código de programación de MySQL. Es un sistema de administración de bases de datos relacional (RDBMS). Se trata de un programa capaz de almacenar una enorme cantidad de datos de gran variedad y de distribuirlos para cubrir las necesidades de cualquier tipo de organización.

MySQL incluye todos los elementos necesarios para instalar el programa, preparar diferentes niveles de acceso de usuario, administrar el sistema y proteger y hacer volcados de datos. Puede desarrollar sus propias aplicaciones de base de datos en la mayor parte de los lenguajes de programación utilizados en la actualidad y ejecutarlos en casi todos los sistemas operativos, MySQL utiliza el lenguaje de consulta estructurado (SQL). Se trata del lenguaje utilizado por todas las bases relacionales, este lenguaje permite crear bases de datos, así como agregar, manipular y recuperar datos en función de criterios específicos.

4.1 ¿Qué es MySQL?

MySQL, el sistema de gestión de base de datos SQL de código abierto más popular, se desarrolla, distribuye, y es apoyado por Oracle Corporation

MySQL es un sistema de gestión de base de datos.

Como se explicó en los primeros capítulos una base de datos es una colección estructurada de datos. Puede ser cualquier cosa, desde una simple lista de compras para una galería de imágenes o las grandes cantidades de información en una red corporativa. Para añadir, acceder y procesar datos almacenados en una base de datos, se necesita un sistema de gestión de bases de datos tales como el servidor MySQL. Dado que los ordenadores son muy buenos en el manejo de grandes cantidades de datos, los sistemas de gestión desempeñan un papel central en la informática, como programas independientes, o como parte de otras aplicaciones.

MySQL relacional.

Las estructuras de bases de datos están organizadas en archivos físicos optimizados para la velocidad. Los modelos lógicos, con objetos tales como bases de datos, tablas, vistas, filas y columnas, ofrece una solución flexible entorno de programación. Configurando así las reglas que rigen las relaciones entre los diferentes datos, campos, como uno-a-uno, uno-a-muchos, único, obligatorio u opcional, y "punteros" entre sus diferentes tablas. La base de datos hace cumplir estas normas, de manera que con una base de datos bien diseñada.

La parte de SQL de "MySQL" es sinónimo de "Structured Query Language" (Lenguaje de consultas estructurado). SQL es el lenguaje común más normalizado utilizado para bases de datos de acceso. Dependiendo de su entorno de programación, puede introducir SQL directamente (por ejemplo, para generar informes), sentencias SQL incrustar en el código escrito en otro idioma, o utilizar una API específica del idioma que esconde la sintaxis SQL.

4.2 Licencia

Tal como expresáramos al presentar este capítulo MySQL, este servidor de bases de datos se distribuye bajo los términos de la Licencia Publica General GNU, pero no es totalmente hostil al "payware" (software comercial) ya que hay una licencia comercial disponible para aquellos que quieran distribuir aplicaciones no GPL que requieran MySQL.
Se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso.

El software es desarrollado por una comunidad pública y los derechos de autor del código están en poder del autor individual, MySQL es patrocinado por una empresa privada, que posee el copyright de la mayor parte del código. Esto es lo que posibilita el esquema de licenciamiento anteriormente mencionado. Además de la venta de licencias privativas, la compañía ofrece soporte y servicios. (LA LICENCIA DE MYSQL)

4.3 MySQL es muy rápido, fiable, escalable y fácil de usar.

Son muchas las razones para escoger MySQL entre ellas:

- **Coste**: El coste de MySQL es gratuito para la mayor parte de los usos y su servicio de asistencia resulta económico.
- **Asistencia**: MySQL AB ofrece contratos de asistencia a precios razonables y existe una nutrida y activa comunidad MySQL.
- **Velocidad**: MySQL es muchas más rápido que la mayor parte de su competencia.
- **Funcionalidad**: MySQL dispone de muchas de las funciones que exigen los desarrolladores profesionales, como compatibilidad completa con ACID, compatibilidad para la mayor parte de SQL ANSI, volcados online, duplication, funciones SSL, e integración con la mayor parte de los entornos de programación. Así mismo, se desarrolla y actualiza de forma mucho más rápida, por lo que prácticamente todas las funciones estándar de MySQL todavía no están en fase de desarrollo.
- **Portabilidad**: MySQL se ejecuta en la inmensa mayoría de sistema operativos y la mayor parte de los casos, los datos se pueden transferir de un sistema a otro sin dificultad.
- **Facilidad de uso**: MySQL resulta fácil de utilizar de administrar. Gran parte de las viejas bases de datos presentan problemas por utilizar sistemas obsoletos, lo que complica innecesariamente las tareas de administración. Las herramientas de MySQL son potentes y flexibles, sin sacrificar su capacidad de uso.

4.4 Las principales características de MySQL.

En esta sección se describen algunas de las características más importantes del software de base de datos MySQL:

- Escrito en C y C ++.
- Usa GNU Automake, Autoconf, y Libtool para portabilidad
- Uso de multihilos mediante hilos del kernel.

- Usa tablas en disco b-tree para búsquedas rápidas con compresión de índice
- Tablas hash en memoria temporales
- El código MySQL se prueba con Purify (un detector de memoria perdida comercial) así como con Valgrind, una herramienta GPL.
- Completo soporte para operadores y funciones en cláusulas select y where.
- Completo soporte para cláusulas group by y order by, soporte de funciones de agrupación
- Seguridad: ofrece un sistema de contraseñas y privilegios seguro mediante verificación basada en el host y el tráfico de contraseñas está cifrado al conectarse a un servidor.
- Soporta gran cantidad de datos. MySQL Server tiene bases de datos de hasta 50 millones de registros.
- Se permiten hasta 64 índices por tabla (32 antes de MySQL 4.1.2). Cada índice puede consistir desde 1 hasta 16 columnas o partes de columnas. El máximo ancho de límite son 1000 bytes (500 antes de MySQL 4.1.2).
- Los clientes se conectan al servidor MySQL usando sockets TCP/IP en cualquier plataforma. En sistemas Windows se pueden conectar usando named pipes y en sistemas Unix usando ficheros socket Unix.
- En MySQL 5.0, los clientes y servidores Windows se pueden conectar usando memoria compartida.
- MySQL contiene su propio paquete de pruebas de rendimiento proporcionado con el código fuente de la distribución de MySQL.

Inicialmente, MySQL carecía de elementos considerados esenciales en las bases de datos relacionales, tales como integridad referencial y transacciones. A pesar de ello, atrajo a los desarrolladores de páginas web con contenido dinámico, justamente por su simplicidad.

Poco a poco los elementos de los que carecía MySQL están siendo incorporados tanto por desarrollos internos, como por desarrolladores de software libre. Entre las características disponibles en las últimas versiones se puede destacar:

- Amplio subconjunto del lenguaje SQL. Algunas extensiones son incluidas igualmente.
- Disponibilidad en gran cantidad de plataformas y sistemas.
- Posibilidad de selección de mecanismos de almacenamiento que ofrecen diferentes velocidades de operación, soporte físico, capacidad, distribución geográfica, transacciones...
- Transacciones y claves foráneas.
- Conectividad segura.
- Replicación.
- Búsqueda de indexación de campos de texto.

4.4.1 Tipos de datos

Este gestor Utiliza muchos tipos de datos: signed/unsigned integers 1, 2, 3, 4, y 8 bytes long, FLOAT, DOUBLE, CHAR, VARCHAR, BINARY, VARBINARY, TEXT, BLOB, DATE, TIME, DATETIME, TIMESTAMP, YEAR, SET, ENUM, y OpenGIS tipos espaciales.

4.5 ¿Las declaraciones y Funciones?

MySQL es un sistema de administración de base de datos relacional. Lógicamente, los datos se estructuran en tablas, que se relacionan entre sí por un campo común. Las tablas se componen de filas (o registros) y los registros se componen de columnas (o campos). Los campos pueden ser de diferente tipo: numéricos, de cadena o de tipo fecha.

El servidor MySQL es el lugar en el que se almacenan los datos y sobre él se ejecutan las consultas. Para establecer una conexión al servidor MySQL, necesita el cliente MySQL. Este puede estar instalado en el mismo equipo que el servidor o en un equipo remoto.

El potencial de un sistema de un sistema de administración de bases de datos procede de su capacidad para estructurar datos y recuperarlos en función de una gran variedad de requisitos específicos. El estándar de la industria para, manipular y definir datos es SQL. Sus comandos más importantes son los siguientes:

- La instrucción **CREATE** crea bases de datos y tablas dentro de la base de datos.
- La instrucción **INSERT** coloca los registros en una tabla.
- La instrucción **SELECT** devuelve los resultados de una columna.
- La instrucción **UPDATE** modifica los datos de una tabla.
- La instrucción **ALTER** cambia la estructura de una tabla, utilizando clausulas como ADD para agregar una nueva columna, CHANGE para cambiar el nombre o definición de una columna existente, RENAME para cambiar el nombre de una tabla o DROP para eliminar una tabla.

Las funciones incrementan el potencial de MySQL. Las funciones se caracterizan por ir seguidas de paréntesis. MySQL incorpora una gran cantidad de funciones (matemáticas, como SUM () para calcular el total de una conjunción de fecha y hora, como YEAR () para extraer la porción del año de una fecha, y funciones de cadena, como RIGHT () para extraer parte de una cadena que comience por el lado derecho de dicha cadena). (GILFILLAN, LA BIBLIA DE MYSQL, 2003)

4.6 Lenguajes de programación

Existen varias interfaces de programación de aplicaciones que permiten, a aplicaciones escritas en diversos lenguajes de programación, acceder a las bases de datos MySQL, incluyendo C, C++, C#, Pascal, Delphi (vía dbExpress), Eiffel, Smalltalk, Java (con una implementación nativa del driver de Java), Lisp, Perl, PHP, Python, Ruby, Gambas, REALbasic (Mac y Linux), (x)Harbour (Eagle1), FreeBASIC, y Tcl; cada uno de estos utiliza una interfaz de programación de aplicaciones específica. También existe una interfaz ODBC, llamado MyODBC que permite a cualquier lenguaje de programación que soporte ODBC comunicarse con las bases de datos MySQL. También se puede acceder desde el sistema SAP, lenguaje ABAP.MS

4.6.1 Aplicaciones

MySQL es muy utilizado en aplicaciones web, como Joomla, Wordpress, Drupal o phpBB, en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece en combinación con MySQL. MySQL es una base de datos muy rápida en la lectura cuando utiliza el motor no transaccional MyISAM, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones. Sea cual sea el entorno en el que va a utilizar MySQL, es importante monitorizar de antemano el rendimiento para detectar y corregir errores tanto de SQL como de programación.

4.7 SEGURIDAD

Una gran cantidad de usuarios y clientes que confían en MySQL como solución tecnológica a sus necesidades de bases de datos, de ahí que muchos sitios web, blogs, ecommerce y webapps cuenten con esta base de datos

- Un sistema de privilegios y contraseñas que es muy flexible y seguro, y que permite basado en host de verificación.
- La seguridad de contraseña de cifrado de todo el tráfico de la contraseña cuando se conecta a un servidor.

Así que existen 5 buenas prácticas de seguridad que ayudarán a ser más "fuerte" a la base de datos y estar mejor optimizada y preparada ante cualquier posible ataque externo:

1. Comprobar que has "blindado" las posibilidades de que usuarios externos puedan inyectar código a tu base de datos a través de formularios o campos de texto públicos de tu sitio web.

2. Cambiar el usuario root, que viene por defecto, y asigna otro nombre de usuario.

3. Asegurar que la contraseña de root de MySQL se ha establecido.

4. Eliminar la cuenta de prueba y base de datos de prueba que se ha creado durante la instalación inicial de MySQL.

5. Revisar periódicamente los usuarios y bases de datos MySQL de una cuenta para asegurarte que los permisos que les otorgaste en su momento permanezcan exactamente como los dejaste la última vez.

MySQL proporciona una gran cantidad de elementos de seguridad, mismos que, claro está, están a nuestra total disposición. Una de estas bendiciones de seguridad son las garantías de uso para diferentes usuarios. Podemos definir varios usuarios para nuestras bases de datos, y a todos y cada uno podemos garantizarles, negarles o revocarles ciertos privilegios para el manejo de nuestras bases de datos.

Una sana recomendación es, precisamente, permitir el "abuso" total para con nuestras bases, única y exclusivamente al usuario root (que vendría a ser algo así como el superadministrador). Para garantizar los privilegios a un usuario específico, tenemos la sentencia **GRANT**, que es precisamente la que le dice a MySQL si tal o cual usuario tienen o no tiene los privilegios para hacer algo con nuestras bases de datos. Ese algo podría ser la habilidad de hacer consultas (**SELECT**), la habilidad de actualizar o modificar (**UPDATES**), la habilidad de insertar información (**INSERT**) o la habilidad de borrar (**DELETE**).

La Instrucción **GRANT** es utilizada una vez que iniciamos una sesión MySQL con el usuario **ROOT**, es decir, cuando el comando del sistema escribimos lo siguiente:

mysql –u root

Una vez dentro de la sesión con el usuario raíz o superadministrador, debemos utilizar la sentencia GRANT de la siguiente manera:

GRANT privilegio_a_otorgar ON nombre_base_de_datos TO usuario@localhost IDENTIFIED BY password

Posteriormente podemos probar estos privilegios entrando a otra sesión de MySQL con el usuario al cual le hemos permitido los privilegios e intentar hacer algo que no le hemos permitido a dicho usuario.

Un ejemplo, una vez estemos dentro del usuario **ROOT**:

GRANT SELECT ON prueba.* TO usuario1@localhost INDENTIFIED BY 'dejameentrar';

Para probar esto se debe salir de la sesión **ROOT** simplemente digitando **EXIT** en la línea de comandos de MySQL y entrando luego al usuario específico digitando lo siguiente:

mysql –u usuario1 –p

De esta manera se ha iniciado una sesión con el usuario **USUARIO1**. La sentencia **GRANT** anterior le dice a MySQL que:

Garantizar el privilegio de SELECCIONAR EN prueba EN **usuario@localhost**lDENTIDICADO POR 'dejameentrar';

La directiva **–p** que se ha especificado al entrar en la sesión usuario1 hace que MySQL pide una contraseña antes de entrar, recordando que la contraseña especificada fue 'dejameentrar', así que se escribe precisamente dejameentrar al momento de entrar a la sesión.

Una vez dentro, al estar en la línea de comandos de MySQL (mysql>) podemos intentar, por ejemplo, algo como esto:

DELETE FROM prueba;

Lo que MySQL nos dirá es que hay un error, un **error 1142** para ser más exactos, cada vez que intentemos hacer algo para lo que nuestro usuario no tiene usuario (recuerden por favor que usuario1 sólo tiene permiso para hacer consultas (**SELECT**).

Finalmente, se solicita a MySQL los privilegios de un usuario específico por medio de la sentencia **SHOW GRANTS**, tal y como lo muestra la imagen 5:

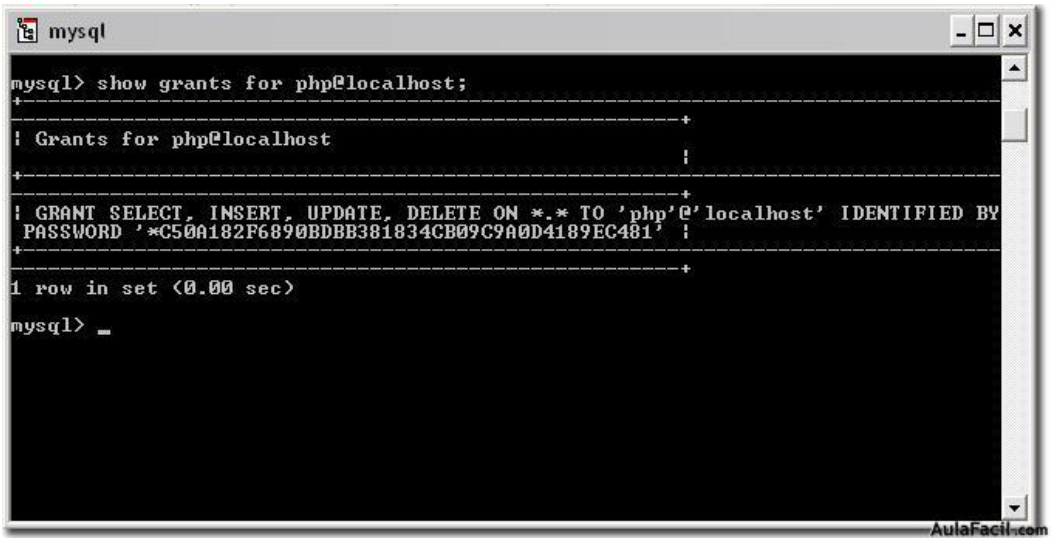

Imagen 5 *Privilegios de un usuario específico*

Esta es una herramienta poderosa en cuanto a la seguridad de nuestras bases de datos se refiere.

4.8 Escalabilidad y Límites

- Soporte para grandes bases de datos. Utilizamos MySQL Server con bases de datos que contienen 50 millones de registros.
- También se sabe de usuarios que usan MySQL Server con 200.000 mesas y unos 5000 millones de filas.

4.9 Conectividad

Los clientes pueden conectarse al servidor MySQL usando varios protocolos:

- Los clientes pueden conectarse a través de sockets TCP / IP en cualquier plataforma.
- En los sistemas Windows, los clientes pueden conectarse a través de canalizaciones con nombre si el servidor se inicia con el --enable-named-pipe. Los servidores Windows soportan conexiones con memoria compartida si iniciado con la opción --shared-memory. Los clientes pueden conectar a través de memoria compartida utilizando el –protocol-memory de memoria.

- En los sistemas Unix, los clientes pueden conectarse a través de ficheros socket Unix.
- Los programas cliente MySQL pueden escribirse en muchos idiomas. Una biblioteca de cliente escrito en C está disponible para clientes escritos en C o C ++, o para cualquier lenguaje que ofrece enlaces de C.
- API para C, C ++, Eiffel, Java, Perl, PHP, Python, Ruby y Tcl están disponibles, permitiendo a MySQL clientes para ser escritos en muchos idiomas.

4.10 REQUERIMIENTOS

Antes de instalar cualquier SGBD es necesario conocer los requerimientos de hardware y software, el posible software a desinstalar previamente, verificar el registro de Windows y el entorno del sistema, así como otras características de configuración especializadas como pueden ser la reconfiguración de los servicios TCP/IP y la modificación de los tipos archivos HTML para los diversos navegadores. Se presenta a continuación una serie de requerimientos mínimos de hardware y software para instalar MySQL estándar versión 5.1 en Windows Seven.

La regla general para determinar el tamaño de la memoria virtual depende del tamaño de memoria RAM instalada. Si el sistema tiene menos de 4 GB de RAM por lo general el espacio de intercambio debe ser de al menos dos veces este tamaño. Si se tiene más de 8 GB de memoria RAM instalada se puede considerar usar el mismo tamaño como espacio de intercambio. Cuanta más memoria RAM tenga instalada, es menos probable usar el espacio de intercambio, a menos que tenga un proceso inadecuado. Requisitos de Windows Para ejecutar MySQL

Para Windows, se necesita lo siguiente:

- Un sistema operativo Windows de 32 bits, tal como 9x, Me, NT, 2000, XP, o Windows Server 2003. Se recomienda fuertemente el uso de un sistema operativo Windows basado en NT (NT, 2000, XP, 2003) puesto que éstos permiten ejecutar el servidor MySQL como un servicio
- Soporte para protocolo TCP/IP.
- Una copia de la distribución binara de MySQL para Windows. (Morales, 2014)

Capítulo 5 POSTGRESQL

El Sistema Gestor de Bases de Datos Relacionales Orientadas a Objetos conocido como PostgreSQL (y brevemente llamado Postgres95) está derivado del paquete Postgres escrito en Berkeley. Con cerca de una década de desarrollo tras él, PostgreSQL es el gestor de bases de datos de código abierto más avanzado hoy en día, ofreciendo control de concurrencia multi-versión, soportando casi toda la sintaxis SQL (incluyendo subconsultas, transacciones, y tipos y funciones definidas por el usuario), contando también con un amplio conjunto de enlaces con lenguajes de programación (incluyendo C, C++, Java, perl, tcl y python).

Postgres, desarrollada originalmente en el Departamento de Ciencias de la Computación de la Universidad de California en Berkeley, fue pionera en muchos de los conceptos de bases de datos relacionales orientadas a objetos que ahora empiezan a estar disponibles en algunas bases de datos comerciales. Ofrece suporte al lenguaje SQL: 2003, integridad de transacciones, y extensibilidad de tipos de datos. PostgreSQL es un descendiente de dominio público y código abierto del código original de Berkeley. (Martinez Guerrero)

5.1 ¿Qué es PostgreSQL?

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional (ORDBMS) basado en Postgres, pionero de muchos conceptos que sólo estuvo disponible en algunos sistemas de bases de datos comerciales.

PostgreSQL es un descendiente de código abierto del código original de Berkeley. Es compatible con una gran parte de la Estándar SQL y ofrece muchas características modernas:

- Consultas complejas
- Claves externas
- Disparadores
- Vistas

- Integridad transaccional
- Control de concurrencia multiversión

También, PostgreSQL se puede extender por el usuario en muchas maneras, por ejemplo mediante la adición de nuevo

- Funciones
- Operadores
- Funciones de agregado
- Métodos de índice
- Idiomas de procedimiento

Y debido a la licencia liberal, PostgreSQL puede ser utilizado, modificado y distribuido por todo el mundo gratis de forma gratuita para cualquier fin, ya sea privada, comercial o académico. (Martinez Guerrero)

5.2 Una breve historia de PostgreSQL

El sistema de gestión de bases de datos objeto-relacional ahora conocido como PostgreSQL se deriva de los POSTGRES paquete escrito en la Universidad de California en Berkeley. Con más de una década de desarrollo detrás de él, PostgreSQL es ahora la base de datos de código abierto más avanzada disponible en cualquier lugar. (Guerrero)

5.2.1 El Proyecto Berkeley POSTGRES

El proyecto Postgres, liderado por el profesor Michael Stonebraker, fue patrocinado por la Defense Advanced Agencia de Proyectos de Investigación (DARPA), la Oficina de Investigación del Ejército (ARO), la Fundación Nacional para la Ciencia (NSF), y ESL, Inc. La implementación de POSTGRES comenzó en el año de 1986. Los conceptos iniciales para el sistema se presentaron en el diseño de Postgres, y la definición del modelo de datos inicial apareció

La lógica y la arquitectura del gestor de almacenamiento fueron detalladas en el diseño del sistema de almacenamiento de Postgres. POSTGRES ha sido objeto de varios lanzamientos

importantes desde entonces. El primer sistema de "Demoware" entró en funcionamiento en 1987 y se presentó en la Conferencia ACM-SIGMOD 1988. Versión 1, fue puesto en libertad a algunos usuarios externos en junio de 1989. En respuesta a una crítica del primer sistema de reglas, y la versión 2 fue lanzado en junio de 1990 con el nuevo sistema de reglas. La versión 3 apareció en 1991 y ha añadido soporte para múltiples gestores de almacenamiento, un ejecutor de consultas mejorado, y un sistema de reglas reescrito. En su mayor parte, las versiones posteriores hasta Postgres95 se centró en la portabilidad y fiabilidad.

POSTGRES se ha utilizado para poner en práctica muchas diferentes aplicaciones de investigación y producción. Éstas incluyen: un sistema financiero análisis de datos, un paquete de monitorización del rendimiento del motor a reacción, un seguimiento de asteroides base de datos, una base de datos la información médica, y varios sistemas de información geográfica. Postgres también se ha utilizado como herramienta educativa en varias universidades. Finalmente, la información Illustra Tecnologías (más tarde fusionado en Informix2, que ahora es propiedad de IBM3) recogido en el código y comercializado. A finales de 1992, se convirtió en el gerente POSTGRES datos primarios para la computación científica Sequoia 2000 proyecto4. El tamaño de la comunidad de usuarios externa casi se duplicó durante 1993. Se hizo cada vez más evidente que el mantenimiento del código del prototipo y el apoyo estaba tomando grandes cantidades de tiempo que debería haber sido dedicado a la investigación de base de datos. En un esfuerzo por reducir esta carga de apoyo, el proyecto Berkeley POSTGRES terminó oficialmente con la versión 4.2. (Martinez Guerrero)

5.2.2 PostgreSQL

Se eligió el nombre de PostgreSQL, para reflejar la relación entre los POSTGRES originales y las versiones más recientes con capacidad de SQL.

El énfasis durante el desarrollo de Postgres95 era en la identificación y comprensión de los problemas existentes en el código del servidor. Con PostgreSQL, el énfasis se ha desplazado a aumentar características y capacidades, aunque se sigue trabajando en todas las áreas. (Guerrero)

5.3 Informe de errores

Los errores en PostgreSQL juegan un papel importante porque incluso el máximo cuidado no puede garantizar que todas las partes del PostgreSQL funcionan en todas las plataformas en toda circunstancia. No se puede arreglar todos los errores de inmediato si el error es obvio, crítico, o afecta a una gran cantidad de usuarios. De ser necesaria la ayuda de inmediato, se debe considerar la obtención de un contrato de soporte comercial. (Martinez Guerrero)

5.3.1 La identificación de errores

Las siguientes circunstancias pueden ser errores de identificación por lo cual no exista un buen desempeño de PostgreSQL

- Un programa termina un mensaje de error del sistema operativo que apuntaría a un problema en el programa.
- Un programa produce la salida equivocada para cualquier entrada dada.
- Un programa se niega a aceptar la entrada válida
- Un programa acepta la entrada inválida y sin previo aviso o error un mensaje. Pero hay que tener en cuenta que su idea de entrada no válida podría ser nuestra idea de una extensión o la compatibilidad con la práctica tradicional.
- PostgreSQL falla al compilar, construir o instalar de acuerdo con las instrucciones que aparecen en las plataformas soportadas. (Martinez Guerrero)

5.4 Fundamentos de la Arquitectura

En el lenguaje la base de datos PostgreSQL utiliza un modelo cliente / servidor. Una sesión de PostgreSQL consiste en lo siguientes procesos:

- **Un proceso de servidor**, que gestiona los archivos de base de datos, acepta las conexiones a la base de datos desde el cliente aplicaciones y realiza acciones sobre la base de datos en nombre de los clientes. El programa servidor de base se llama postgres.
- **El usuario cliente** que quiere llevar a cabo las operaciones de base de datos. Las aplicaciones cliente puede ser de naturaleza muy diversa: un cliente podría ser una

herramienta orientada a texto, una aplicación gráfica, un servidor web que accede a la base de datos para mostrar las páginas web, o una herramienta especializada de mantenimiento de bases de datos. Algunos clientes aplicaciones se suministran con la distribución de PostgreSQL; la mayoría son desarrollados por los usuarios.

Como es típico de aplicaciones cliente / servidor, el cliente y el servidor pueden estar en diferentes hosts. El servidor PostgreSQL puede manejar múltiples conexiones simultáneas de clientes. Para ello se inicia ("forks") un nuevo proceso para cada conexión. A partir de ese punto el cliente y el nuevo proceso de servidor pueden comunicarse sin intervención por el proceso postgres originales. Por lo tanto, el proceso de servidor maestro es siempre corriendo, esperando conexiones de cliente, mientras que el cliente y el servidor se asocian al proceso. (Martinez Guerrero)

5.4.1 CARACTERÍSTICAS

Los sistemas de mantenimiento de Bases de Datos relacionales tradicionales (DBMS) soportan un modelo de datos que consisten en una colección de relaciones con nombre, que contienen atributos de un tipo específico. Postgres ofrece una potencia adicional sustancial al incorporar los siguientes cuatro conceptos adicionales básicos en una vía en la que los usuarios pueden extender fácilmente el sistema

- Tipos de datos
- Funciones
- Operadores
- Funciones de agregado
- Métodos de indexación
- Lenguajes procedurales

Otras características aportan potencia y flexibilidad adicional:

- Consultas complejas (subconsultas, joins, etc.)
- Integridad referencial (claves primarias y foráneas)

- Restricciones (Constraints)
- Disparadores (triggers)
- Vistas (views)
- Reglas (rules)
- Integridad transaccional
- Control de concurrencia multiversión

Estas características colocan a Postgres en la categoría de las Bases de Datos identificadas como objeto-relacionales. Nótese que éstas son diferentes de las referidas como orientadas a objetos, que en general no son bien aprovechables para soportar lenguajes de Bases de Datos relacionales tradicionales. Postgres tiene algunas características que son propias del mundo de las bases de datos orientadas a objetos. De hecho, algunas Bases de Datos comerciales han incorporado recientemente características en las que Postgres fue pionera. (Martinez Guerrero)

Además, debido a su licencia liberal, PostgreSQL puede ser usado, modificado, y distribuido por cualquier persona para cualquier propósito sin cargo alguno, ya sea para un fin privado, comercial o académico. (Grupos de usuarios postgresq de argentina)

Sus características técnicas la hacen una de las bases de datos más potentes y robustos del mercado. Su desarrollo comenzó hace más de 16 años, y durante este tiempo, estabilidad, potencia, robustez, facilidad de administración e implementación de estándares han sido las características que más se han tenido en cuenta durante su desarrollo. PostgreSQL funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema.

Las características más importantes y soportadas por PostgreSQL:

- Es una base de datos 100% ACID.

- Soporta distintos tipos de datos: además del soporte para los tipos base, también soporta datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes (MAC, IP...), cadenas de bits, etc. También permite la creación de tipos propios.
- Incluye herencia entre tablas, por lo que a este gestor de bases de datos se le incluye entre los gestores objeto-relacionales.
- Copias de seguridad en caliente (Online/hot backups)
- Unicode
- Juegos de caracteres internacionales
- Regionalización por columna
- Multi-Version Concurrency Control (MVCC)
- Múltiples métodos de autentificación
- Acceso encriptado vía SSL
- SE-postgres
- Completa documentación
- Licencia BSD
- Disponible para Linux y UNIX en todas sus variantes (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64) y Windows 32/64bit. (Martinez Guerrero)

Alta concurrencia

Mediante un sistema denominado MVCC (Acceso concurrente multiversión, por sus siglas en inglés) PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Cada usuario obtiene una visión consistente de lo último a lo que se le hizo commit. (Martinez Guerrero)

5.4.2 Limitaciones

- Puntos de recuperación dentro de transacciones. Actualmente, las transacciones abortan completamente si se encuentra un fallo durante su ejecución.
- No soporta tablespaces para definir dónde almacenar la base de datos, el esquema, los índices, etc. (versiones antes de la 9.0)
- El soporte a orientación a objetos es una simple extensión que ofrece prestaciones como la herencia, no un soporte completo. (Martinez Guerrero)

5.5 Ventajas:

- Ampliamente popular - Ideal para tecnologías Web.
- Fácil de Administrar.
- Su sintaxis SQL es estándar y fácil de aprender.
- Footprint bajo de memoria, bastante poderoso con una configuración adecuada.
- Multiplataforma.
- Capacidades de replicación de datos.
- Soporte empresarial disponible. (Martinez Guerrero)

5.5.1 Estabilidad y confiabilidad

En contraste a muchos sistemas de bases de datos comerciales, es extremadamente común que compañías reporten que PostgreSQL nunca ha presentado caídas en varios años de operación de alta actividad. Ni una sola vez. Simplemente funciona. (Martinez Guerrero)

5.5.2 Extensible

El código fuente está disponible para todos sin costo. Si su equipo necesita extender o personalizar PostgreSQL de alguna manera, pueden hacerlo con un mínimo esfuerzo, sin costos adicionales. Esto es complementado por la comunidad de profesionales y entusiastas de

PostgreSQL alrededor del mundo que también extienden PostgreSQL todos los días. (Martinez Guerrero)

5.5.3 Multiplataforma

PostgreSQL está disponible en casi cualquier Unix (34 plataformas en la última versión estable), y una versión nativa de Windows está actualmente en estado beta de pruebas. (Martinez Guerrero)

5.5.4 Diseñado para ambientes de alto volumen

PostgreSQL usa una estrategia de almacenamiento de filas llamada MVCC para conseguir una mejor respuesta en ambientes de grandes volúmenes. Los principales proveedores de sistemas de bases de datos comerciales usan también esta tecnología, por las mismas razones. (Martinez Guerrero)

5.6 DESVENTAJAS

- En comparación con *MySQL* es más lento en inserciones y actualizaciones, ya que cuenta con cabeceras de intersección que no tiene *MySQL*.
- Soporte en línea: Hay foros oficiales, pero no hay una ayuda obligatoria.
- Consume más recursos que *MySQL*.
- La sintaxis de algunos de sus comandos o sentencias no es nada intuitiva. (Guerrero)

5.7 Seguridad PostgreSQL

Con el fin de no redundar, de mantener la integridad, la disponibilidad, mantener el control de acceso, así como los privilegios y la autenticación de los usuarios y demás funcionalidades de la base de datos y del gestor. Se abarcara acerca de la seguridad que es de

suma importancia para la excelente gestión de la información, es necesario estudiar la seguridad con que este recurso tan valioso que son los datos serán resguardado. (Martinez Guerrero)

5.7.1 INTEGRIDAD

La integridad referencial es una función que está presente en las bases de datos relacionales que garantiza la coherencia de los datos entre relaciones aparejadas. Sin duda alguna es una de las características más importantes que una base de datos nos puede proporcionar y siempre se debería usar para garantizar la integridad de los datos almacenados. La integridad referencial se define con el uso combinado de claves primarias (primary keys) ó claves candidatas (candidate key) y clave foráneas (foreign key). Ambas claves están formadas por valores únicos y las claves o llaves foráneas solo pueden estar asociadas a una primaria con el fin de garantizar la existencia de un solo dato valido y correcto. Se hace necesario contar con una base de datos normalizada para el uso de estas funciones y obtener beneficios como:

- Evitar redundancia en los datos almacenados.
- Evitar problemas de actualización de los datos en las tablas.
    - Proteger la integridad de los datos. (Martinez Guerrero)

5.7.2 ACCESO A LA BASE DE DATOS

Luego de que la base de datos es creada se puede acceder a ella de varias maneras según sea la necesidad y el tipo de usuario veamos:

- Ejecutando el programa monitor de terminal de Postgres (psql) que le permite introducir, editar y ejecutar órdenes SQL de un modo interactivo.
- Ejecutando el PGADMIN con el cual podemos establecer conexión con el servidor y manipular las bases de datos.
- Desarrollando un software en alguno de los muchos lenguajes de programación existentes, el cual tenga las instrucciones necesarias para conectarse con la base de datos.

Utilizando herramientas reporteadoras para conectarnos a la base de datos y extraer datos para su posterior análisis. (Martinez Guerrero)

5.7.3 SUPERUSUARIOS DE LA BASE DE DATOS

Los SuperUsuarios de la base de datos (aquellos que tienen el campo pg_user.usesuper activado) ignoran todos los controles de acceso descritos anteriormente con dos excepciones: las actualizaciones del catálogo del sistema no están permitidas si el usuario no tiene el campo pg_user.usecatupd activado, y nunca se permite la destrucción del catálogo del sistema (o la modificación de sus estructuras). (Martinez Guerrero)

5.7.4 PRIVILEGIOS DE ACCESO

El uso de los privilegios de acceso para limitar la lectura, escritura y la puesta de reglas a las clases se trata en grant/revoke. (Martinez Guerrero)

5.7.5 BORRADO DE CLASES Y MODIFICACIÓN DE ESTRUCTURAS.

Los comandos que borran o modifican la estructura de una clase, como **alter, drop table, y drop index**, solo funcionan con el propietario de la clase. Como hemos dicho antes, estas operaciones no están permitidas nunca en los catálogos del sistema. (Martinez Guerrero)

5.7.6 LA SEGURIDAD DE LA BASE DE DATOS

La seguridad de la base de datos esta implementada en varios niveles:

- Protección de los ficheros de la base de datos. Todos los ficheros almacenados en la base de datos están protegidos contra escritura por cualquier cuenta que no sea la del super usuario de Postgres.
- Las conexiones de los clientes al servidor de la base de datos están permitidas, por defecto, únicamente mediante sockets Unix locales y no mediante sockets TCP/IP. Ha de arrancarse el demonio con la opción -i para permitir la conexión de clientes no locales.
- Las conexiones de los clientes se pueden restringir por dirección IP y/o por nombre de usuario mediante el fichero pg_hba.conf situado en PG_DATA.

- Las conexiones de los clientes pueden ser autentificadas mediante otros paquetes externos.
- A cada usuario de Postgres se le asigna un nombre de usuario y (opcionalmente) una contraseña. Por defecto, los usuarios no tienen permiso de escritura a bases de datos que no hayan creado.

Los usuarios pueden ser incluidos en grupos, y el acceso a las tablas puede restringirse en base a esos grupos. (Martinez Guerrero)

5.7.7 AUTENTICACIÓN DE USUARIOS

La autenticación es el proceso mediante el cual el servidor de la base de datos y el postmaster se aseguran de que el usuario que está solicitando acceso a la base de datos es en realidad quien dice ser. Todos los usuarios que quieren utilizar Postgres se comprueban en la tabla pg_user para asegurarse que están autorizados a hacerlo. Actualmente, la verificación de la identidad del usuario se realiza de distintas formas: (Martinez Guerrero)

- **DESDE LA SHELL DEL USUARIO**

Un demonio (servicio) que se lanza desde la shell del usuario anota el id original del usuario antes de realizar un setuid al id del usuario postgres. El id original del usuario se emplea como base para todo tipo de comprobaciones.

- **DESDE LA RED**

Si Postgres se instala como distribuido, el acceso al puerto TCP del postmaster está disponible para todo el mundo. El ABD configura el fichero pg_hba.conf situado en el directorio PG_DATA especificando el sistema de autenticación a utilizar en base al equipo que realiza la conexión y la base de datos a la que se conecta.

5.7.8 COPIA DE SEGURIDAD Y RESTAURACIÓN

Deben realizarse copias de seguridad de las bases de datos regularmente. Dado que Postgres gestiona sus propios ficheros en el sistema, no se recomienda confiar en los sistemas de copia de seguridad del sistema para las copias de respaldo de las bases de datos; no hay garantía de que los ficheros estén en un estado consistente que permita su uso después de la restauración.

Postgres proporciona dos utilidades para realizar las copias de seguridad de su sistema: pg_dump para copias de seguridad de bases de datos individuales y pg_dumpall para realizar copias de seguridad de toda la instalación de una sola vez. La copia de seguridad de una sola base de datos puede realizarse usando la siguiente orden: (Martinez Guerrero)

- **% pg_dump nombredb > nombredb.pgdump**

y puede ser restaurada usando

- **cat nombredb.pgdump | psql nombredb**

Esta técnica puede usarse para mover bases de datos a una nueva localización y para renombrar bases de datos existentes. (LOPEZ)

Capítulo 6 COMPARACIÓN DE MySQL Y PostgreSQL

La Tabla 4 muestra las propiedades de los gestores respecto a los criterios de comparación: la velocidad en respuesta, el modelo conceptual, tipos de datos soportados, la fiabilidad así como la extensibilidad, la arquitectura de implementación, manejo de transacción, concurrencia, tamaño de registros, respaldo de base de datos, permisos de usuarios, disponibilidad de drivers para JDBC, interfaz y plataforma soportada entre los Sistemas Gestores de Base de Datos MySQL y PostgreSQL, teniendo una visión más amplia de lo que los hace diferentes y los caracteriza, haciéndolos mejor en el desempeño de un área específica.

Tabla 4 Tabla comparativa de MySQL y PostgreSQL

Criterio	MYSQL	POSTGRESQL
Velocidad	MySQL es mucho más rápido que la mayor parte de su competencia ya que tiene la posibilidad de selección de mecanismos de almacenamiento que ofrecen diferentes velocidades de operación, soporte físico, capacidad, distribución geográfica, transacciones. Aprovecha la potencia de sistemas multiprocesador, gracias a su implementación multihilo.	La velocidad de respuesta que ofrece este gestor con bases de datos relativamente pequeñas puede parecer un poco deficiente, aunque esta misma velocidad la mantiene al gestionar bases de datos realmente grandes, cosa que resulta loable. Es capaz de ajustarse al número de CPUs y a la cantidad de memoria que posee el sistema de forma óptima, haciéndole capaz de soportar una mayor cantidad de peticiones simultáneas de manera correcta, se dice que ha llegado a soportar el triple de carga de lo que soporta MySQL. Multiprocesos.
Modelo Conceptual	Modelo relacional	Orientado a objetos

Tipo de Datos soportados	Tipos numéricos: TinyInt, Bit ó Bool, SmallInt, SmallInt, MediumInt, Integer Int, BigInt, Float, Double, Decimal o Numeric. Tipos de Fecha: Date, DateTime, TimeStamp, Time, Year. Tipos de Cadena: Char(n), VarChar(n), TinyText y TinyBlob, Blob y Text, MediumBlob y MediumText, LongBlob y LongText, Enum, Set.	Soporta tipos complejos como registros, conjuntos, referencias, listas, pilas, colas y arreglos. Tipos Numéricos: smallint, integer, bigint, decimal, numeric, real, double precisión, serial, bigserial. Tipos Monetarios: money Tipos de caracteres: character varying(n), varchar(n), character(n), char(n), text. Tipos de datos binarios: bytea Tipos de fecha y hora: timestamp [(p)] [without time zone], timestamp [(p)] with time zone, time [(p)] [without time zone], time [(p)] with time zone, interval [fields] [(p)]. Tipos Booleanos: TRUE (t, true, y, yes, on, 1). FALSE (f, false, n, no, off, 0).
Fiabilidad	MySQL resulta fácil de utilizar y administrar. Las herramientas de MySQL son potentes y flexibles, sin sacrificar su capacidad de uso.	Sus características técnicas la hacen una de las bases de datos más potentes.
Extensibilidad	Cualquier persona puede bajar el código fuente de MySQL y usarlo sin pagar. Cualquier interesado puede estudiar el código fuente y ajustarlo a sus necesidades. Ya que el código fuente está disponible para todos sin costo.	El código fuente está disponible para todos sin costo. Si su equipo necesita extender o personalizar PostgreSQL de alguna manera, pueden hacerlo con un mínimo esfuerzo, sin costos adicionales.
Arquitectura de Implementación	Cliente / servidor	Cliente / servidor

Manejo de transacciones	MySQL cuenta con la agrupación de transacciones, reuniendo múltiples transacciones de varias conexiones para incrementar el número de transacciones por segundo.	Limitación: Actualmente, las transacciones abortan completamente si se encuentra un fallo durante su ejecución.
Concurrencia	Alta concurrencia: MySQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos.	Alta concurrencia: PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Cada usuario obtiene una visión consistente de lo último a lo que se le hizo commit.
Tamaño en registros	MySQL tiene bases de datos de hasta 50 millones de registros.	Ilimitado (Depende del sistema de almacenamiento).
Respaldo de Bases de Datos	MySQL cuenta con comandos específicos para copia de respaldo y también restaurar la base de datos si esta llegase a perderse, aunque no siempre sea la mejor opción ya que ha llegado a fallar en pocas ocasiones.	Postgresql cuenta con comandos específicos para la copia de respaldo y también la restauración de la base de datos si esta llegase a perderse.
Permisos de usuarios	Ofrece un sistema de contraseñas y privilegios seguro mediante verificación basada en el host y el tráfico de contraseñas está cifrado al conectarse a un servidor.	Manteniendo un sistema de contraseñas desde hace 16 años así como privilegios y la máxima seguridad para el sistema.
Disponibilidad de Drivers para JDBC	Disponible en : https://dev.mysql.com/downloads/connector/j/5.0.html Nombre de Clase Driver: "com.mysql.jdbc.Driver" Cadena de Conexión: jdbc:mysql://localhost:3306/b1s	Disponible en https://jdbc.postgresql.org/ Nombre de la Clase Driver "org.postgresql.Driver"; Cadena de Conexión para posgresql "jdbc:postgresql://localhost:5432/Taller";

Conexiones con .NET	ADO.NET ActiveX Data Objects (ADO). Es un conjunto de componentes del software para acceder a datos y a servicios de datos. Es comúnmente usado para acceder y para modificar los datos almacenados en un Sistema Gestor de Bases de Datos Relacionales. Connector/Net es un conductor ADO.NET completamente gestionada para MySQL. Cadena de conexión para MySQL: cnn.Open "Provider=MySQLProv;" & "Data Source=base;" & _ "User Id=USUARIO;Password=PASSWORD"	ADO.NET es fácil de usar ya que el asistente de aplicación le permite construir la cadena de conexión ADO, seleccionar tablas, campos e índices de conversión, ver y editar secuencias de comandos SQL para generar la base de datos PostgreSQL de destino y seleccionar tablas para la importación. Cadena de conexión para PostgreSQL: Provider=MSDASQL;Driver={PostgreSQL ANSI};SERVER=localhost;DATABASE=saraza;UID=sistema;PWD=saraza;
Interfaz	Existen varias interfaces de programación de aplicaciones que permiten, a aplicaciones escritas en diversos lenguajes de programación, acceder a las bases de datos MySQL, incluyendo C, C++, C#, Pascal, Delphi.	La instalación de PostgreSQL incluye sólo C y la interfaces embebidas de C.
Plataforma soportada	Multiplataforma. MySQL cuenta con disponibilidad en gran cantidad de plataformas y sistemas, tales como; GNU/Linux. Y en sistemas operativos de Microsoft Windows. Usa GNU Automake, Autoconf, y Libtool para portabilidad	Cualquier plataforma moderna tipo Unix debe ser capaz de ejecutar PostgreSQL, también corre de forma nativa en sistemas operativos basados en Microsoft Windows NT como Win2000 SP4, WinXP y Win2003. Multiplataforma

CONCLUSIONES

Este proyecto se basó en la necesidad de proveer información detallada sobre las características más importantes de los Sistemas Gestores de Bases de datos: MySQL y PostgreSQ. Ya que son software muy complejo y sofisticado y no pueden generalizarse sobre los elementos que componen a cada uno de ellos, ya que varía uno del otro.

Sin embargo evaluando sus propiedades, ventajas y desventajas y como se relacionan es de mucha utilidad para así tener un conocimiento y una fuente de información más detallada para la toma de decisiones.

Se realizó una investigación consultando diferentes fuentes de información, mediante un estudio comparativo entre estos dos sistemas gestores de bases de datos, con el objetivo de buscar el gestor con mejor desempeño y los mejores requerimientos de cada uno de estos.

Ambos gestores tiene sus grandes cualidades, en el caso de el SGBD MySQL ha sido un gestor de datos muy útil desde que fue creado y con el tiempo se le fueron añadiendo nuevas funciones convirtiéndolo en uno de los más utilizados a nivel mundial ya que soporta una gran cantidad de datos, la velocidad al realizar las operaciones, convirtiéndolo en un gestor con un buen rendimiento. Su conectividad, velocidad y seguridad lo hacen altamente apropiado para acceder a bases de datos.

Las características técnicas PostgreSQL lo hacen una de las bases de datos más potentes y robustos de todos los tiempos, cuenta con estabilidad y potencia. Funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios. Pero no todo son beneficios ya que PostgreSQL en comparación con MySQL es más lento en inserciones y actualizaciones, ya que cuentan con cabeceras de intersección que no tiene MySQL, por lo que consume más recursos que MySQL. Respecto a la documentación de ambos gestores es más fácil encontrar soporte para MySQL que para PostgrsSQL

Con base en el supuesto establecido en la investigación se concluye que MySQL tiene mejor desempeño en entornos web pero tiene desventajas en el manejo de transacciones concurrentes,

por los que se observa que antes de seleccionar un gestor de bases de datos, se deben considerar los requerimientos del sistema a implementar y el tipo de transacciones. Pues con base en ello se podrá seleccionar el más adecuado.

REFERENCIAS

.Theriault , M., & Newman, A. (2002). *Oracle Manual de seguridad.* Mc Graw Hill.

AA., J. D. (s.f.). *Base de Datos* . Obtenido de http://juanjose84.tripod.com/Basedatos.html

Aparicio, S. (s.f.). Obtenido de http://analisis692.blogspot.mx/2009/05/sistema-sgbd-es-el-software-que-permite.html

aragonesa, P. e.-d. (s.f.). *DEMO E-DUCATIVA CATEDU*. Obtenido de http://e-ducativa.catedu.es/44700165/aula/archivos/repositorio/1000/1080/html/31_clasificacin_de_los_sgbd.html

Cares, M. E. (s.f). *Aplicaciones Informaticas.* Patagonia.

Galeon, J. (s.f.). *http://bdjulian.galeon.com/aficiones1782963.html*.

GILFILLAN, I. (2003). *LA BIBLIA DE MYSQL.* ANAYA MULTIMEDIA.

GILFILLAN, I. (2003). *LA BIBLIA DE MYSQL.* ANAYA MULTIMEDIA.

Grupos de usuarios postgresq de argentina. (s.f.). Obtenido de http://www.postgresql.org.ar/trac/wiki/Documentacion

Guerrero, R. M. (s.f.). *PostgreSQL-es*. Obtenido de http://www.postgresql.org.es/

Gutierrez, D. (Enero de 2009). Seguridad en BD. Venezuela.

http://www.gridmorelos.uaem.mx/~mcruz//cursos/micc/MySQL.pdf. (s.f.).

LA LICENCIA DE MYSQL. (s.f.). Obtenido de http://cv.uoc.edu/web/~pberni/faqs/docs/licencia_mysql.pdf

LOPEZ, I. D. (s.f.). *Byspel*. Obtenido de http://byspel.com/seguridad-de-bases-de-datos-postgresql/

Luis Alberto Casillas Santillán, M. G. (s.f.). *Bases de datos en MySQL*. Obtenido de www.uoc.edu

Marc Gibert Ginestá, O. P. (s.f.). *Bases de datos en PostgreSQL*. Obtenido de www.uoc.edu

Morales, J. L. (2014). *Requerimientos de instalacion*. Obtenido de http://es.slideshare.net/josebunbury/requerimientos-de-instalacion-40526344

Pérez, M. t. (2010). Sistema Gestores de Base de Datos.

Piattini, M., García , F., Garzás, J., & Genero, M. (2008). *Medición y estimación del software: técnicas y métodos para mejorar la calidad y la productividad.* Madrid, España: Alfaomega .

Rafael, M. G. (s.f.). *http://www.postgresql.org.es/documentacion*.

Ramakrishnan, & Gehrke. (2007). *Sistemas de Gestion de Bases de datos.* MC Graw Hill.

Silberschatz. (2007). Fundamentos de bases de datos.

Sistemas de Gestión de Bases de Datos. (s.f.). Obtenido de http://users.dsic.upv.es/~jorallo/docent/BDA/castella/tema3_4x1.pdf

zevde datos. (s.f.). Obtenido de https://tecnologiaeinformaticacji.files.wordpress.com/2013/02/lectura-tipos-de-bases-de-datos.pdf

Zevallos, V. (s.f.). *http://es.slideshare.net/VictorZevallos/comparacion-de-gestores-de-base-de-datos*.